Adaptive Cross Approximation for Electromagnetic Analysis of Superconducting Circuits

by

Ben Abraham Pieter Nel

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Electronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Prof Matthys M. Botha

April 2019

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: April 2019

Abstract

Adaptive Cross Approximation for Electromagnetic Analysis of Superconducting Circuits

Ben Abraham Pieter Nel

Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.

Thesis: MEng (Electronic)

April 2019

Electromagnetic analysis of superconducting integrated circuits is routinely required for inductance extraction. FastHenry is a magnetoquasistatic (MQS) analysis tool suitable for this task. It is based on the partial element equivalent circuit (PEEC), integral equation method, with the structure discretised into hexahedral filaments. FastHenry's multilevel fast multipole algorithm (MLFMA) implementation is especially memory efficient, given certain approximations and algorithmic parameter choices. However, errors are introduced into the matrix representation. This thesis describes the implementation of a multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) inside FastHenry as an alternative to its existing MLFMA solver. The thesis also presents two modified grouping strategies to further improve MLACA-SVD efficiency by compressing interactions between larger groups, while maintaining scaling performance consistent with a valid admissibility condition. MLACA-SVD compresses off-diagonal matrix blocks to a specified error tolerance, based on evaluating selected entries. Quadrature recipes presented in this thesis provide guaranteed accuracy of matrix entry evaluation.

Numerical results for examples of practical interest show that the MLACA-SVD memory scaling versus number of filaments, denoted b, is practically identical to that of FastHenry's MLFMA, and is close to $\mathcal{O}(b \log b)$. The MLACA-SVD requires less memory for the same solution accuracy, and furthermore offers complete control over matrix approximation errors. For the examples considered, it is found to be a more efficient solver. The results of the group merging strategies show that required memory is further reduced by approximately 30%. The MLACA-SVD solver with merging requires about four times less memory than FastHenry's MLFMA, for similar accuracy.

Uittreksel

Aanpassingsvaardige Kruisbenadering vir Elektromagnetiese Analise van Supergeleidende Stroombane

("Adaptive Cross Approximation for Electromagnetic Analysis of Superconducting Circuits")

Ben Abraham Pieter Nel

Departement Elektries en Elektroniese Ingenieurswese,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.

Tesis: MIng (Elektronies)

April 2019

Elektromagnetiese analise van supergeleidende geïntegreerde stroombane word gereeld benodig vir induktansie onttrekking. FastHenry is 'n magnetetoquastistiese (MQS) analisehulpmiddel, geskik vir hierdie taak. Dit is gebaseer op die gedeeltelike element ekwivalente stroombaan (GEES), integrale vergelykings metode, met die struktuur gediskretiseer in hexahedrale filamente. Die multivlak vinnige multipool algoritme (MVVMA) implementering in FastHenry is veral doeltreffend ten opsigte van geheue, gegewe sekere benaderings en algoritmiese parameter keuses. Foute word egter in die matriksvoorstelling ingevoer. Hierdie tesis beskryf die implementering van 'n multivlak aanpassingsvaardige kruisbenadering oplosser met enkelvoudige waarde-ontbinding herkompressie (AKO-EWOH) binne FastHenry as 'n alternatief vir die bestaande MVVMA oplosser. Die tesis bied ook twee aangepaste groeperingstrategieë aan om AKO-EWOH se doeltreffendheid verder te verbeter deur interaksies tussen groter groepe te kompres, terwyl die skaleringsuitset in ooreenstemming bly met 'n geldige toelaatbaarheidstoestand. AKO-EWOH kompres skuinsmatige matriksblokke tot 'n gespesifiseerde fouttoleransie, gebaseer op die evaluering van geselekteerde inskrywings. Kwadratuur resepte wat in hierdie tesis aangebied word, bied gewaarborgde akkuraatheid van matriksinskrywing evaluering.

Numeriese resultate vir voorbeelde van praktiese belang toon dat die skalering van AKO-EWOH se geheue teenoor die aantal filamente, b, feitlik identies is aan dié van FastHenry se MVVMA, en baie naby is aan $\mathcal{O}(b \log b)$. Vir die voorbeelde wat oorweeg is, gebruik AKO-EWOH minder geheue vir dieselfde oplossingsakkuraatheid, en bied bowendien volledige beheer oor matriksbenaderings foute. Die resultate vir die groep samesmelting strategieë toon dat vereiste geheue verder verminder word met ongeveer 30%. Die AKO-EWOH-oplosser met samesmelting verg ongeveer vier keer minder geheue as FastHenry se MVVMA, vir soortgelyke akkuraatheidsvlakke.

Acknowledgements

This paper is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

I am grateful for the guidance provided by my supervisor, Prof Matthys Botha for his insight and enthusiastic support. I would also like to thank Prof Coenraad Fourie for providing me with this opportunity as well as all my colleagues from the superconducting circuit research group at Stellenbosch University.

Above all, I would like to thank my mother for her moral support, encouragement and especially for her unwavering patience.

Contents

Chapter 1

Introduction

1.1 Overview

Superconductivity is a state where conductors have no electrical resistance. This was first described in 1911 by the Dutch physicist H. K. Onnes. Certain materials change to a state of superconductivity at specific temperatures (T), called the critical temperature (T_C) [1]. Computer processors that use superconducting circuits could be significantly more energy-efficient and have greater processing power than conventional ones [2]. A conductor in a normal state has a finite internal magnetic field, however, when T_C is reached, the magnetic field tends to be expelled from the conductor. This is called the Meissner effect. The London penetration depth (λ) describes the penetration depth of the applied magnetic field into the superconductor [1].

The design of superconducting integrated circuits require accurate calculation of the mutual inductances of complicated three-dimensional structures. This behaviour can be simulated with a time domain approach employing a finite element method (FEM) using the Ginzburg-Landau equations [3]. This method generates reliable results in cases where (T) is close to (T_C). However, methods using the FEM discretise the entire three-dimensional volume and when this is done for complex structures, the simulation can require a prohibitively long execution time. A frequency domain, volume integral equation approach can be considered for integrated circuits using the London equations, instead of the full Ginzburg-Landau equations. These equations yield reliable results when T is significantly below T_C.

The process of computing the complex frequency-dependent impedance matrix of a multi-terminal system is referred to as inductance extraction. FastHenry is a magnetoquasistatic (MQS) inductance extraction programme applicable to three-dimensional circuit simulation [4]. Josephson Junctions are used in superconducting circuits as an alternative to CMOS transistors. Inductance extraction of digital circuits is possible by only considering the interconnects, therefore ignoring Josephson Junctions. The current version of FastHenry supports the London penetration depth when determining conductivity. This allows for superconducting circuits to be simulated with FastHenry.

FastHenry requires a structure to be discretised into right-angled hexahedral filaments.

It is based on a volume integral formulation under superconducting MQS assumptions, using the partial element equivalent circuit (PEEC) method to express the electromagnetic problem as a circuit problem [5]. This allows for an elegant mesh current analysis solution, yielding a matrix that is well suited to preconditioning and iterative solution with the GMRES algorithm [6]. With this method, the multilevel fast multipole algorithm (MLFMA) is employed by FastHenry to accelerate the solution. The MLFMA is based on analytical factorisation of the static Green's function and hence, compression of the mutual inductance matrix. It is a well established approach [7, 4].

Superconducting integrated circuits are costly to build and test, thus simulation of these circuits is essential. This can be done with FastHenry for the purposes of inductance extraction. To discretise complex three-dimensional superconducting circuits into right-angled hexahedral filaments, InductEx can be used [8].

In this thesis, the multilevel adaptive cross approximation with singular value decomposition recompression (MLACA-SVD) is newly implemented in FastHenry. The MLACA-SVD is a purely algebraic alternative algorithm to compress mutual inductance matrices. The application of the MLACA-SVD is then compared to the MLFMA in FastHenry. This is possible as both algorithms accelerate the matrix-vector product required within the generalised minimal residual (GMRES) iterative method and reduce the $\mathcal{O}(b^2)$ inductance matrix storage requirement, where b is the number of filaments.

Along with replacing the MLFMA with the MLACA-SVD, this thesis also examines new integration rules for calculating inductance matrix entries [9]. To further reduce the compressed storage of the inductance matrix, a new ACA group merging approach is also presented [10].

It should be noted that there are also FFT-based acceleration (compression) schemes [11, 12], but those require a regular mesh, while the present work is focused on providing an alternative which is directly applicable to FastHenry meshes.

1.2 Project milestones

The main objective of this thesis is to improve FastHenry in terms of accuracy, memory usage and run-time, by implementing the MLACA-SVD as alternative solver to the MLFMA. The milestones towards achieving this main objective are:

1. **Implement ACA inside FastHenry:** The first goal is to implement a single-level ACA version in FastHenry. It is shown how an adiabatic quantum flux parametron (AQFP) gate [13], meshed with 23,226 filaments requires less than three times more storage for single-level ACA using a tolerance of 10^{-3} and ACA applied to all mutual group interactions, than with the standard MLFMA solver. Although applying ACA to bordering groups does not conform to criteria of what may be considered a far interaction [14], the purpose of this work is to show the potential of using ACA instead of MLFMA inside FastHenry. This is presented in [15].

2. **Replace ACA with MLACA:** To obtain competitive compression, it is necessary to implement a multilevel ACA solver. The MLACA is provided with the same group interactions as the MLFMA and it is shown how the memory for both algorithms scales approximately the same between 10,874 and 714,315 filaments with ACA compression tolerances between 10^{-1} and 10^{4}. This is presented in [16].

3. **Add SVD recompression to MLACA:** As the SVD obtains the exact rank of a matrix, it would generate superior compression results. A drawback of the SVD is that it requires knowledge of the entire matrix and therefore would significantly increase the time needed to compress matrices. For this reason SVD is used to recompress ACA compressed matrices to remove any possible redundancies. The SVD tolerance is selected to generate the desired accuracy. This implementation inside FastHenry is presented and evaluated in [9].

4. **Integration recipes for evaluating inductance matrix entries:** FastHenry uses midpoint integration when applying the MLFMA. This restricts the accuracy with which interactions between filaments are calculated. The integration errors are inversely proportional to the distance between filaments. Therefore FastHenry applies the fast multipole method (FMM) to groups that are separated from each other by more than two groups. ACA is based on calculating selected matrix entries. These calculations must be accurate, otherwise the ACA compression tolerance is not a true reflection of the accuracy with which inductance matrix entries are represented. A set of rules (integration recipes) is derived to integrate over right-angled hexahedral filaments relative to distance between them [9]. These integration recipes are targeted at filament aspect ratios almost always present in meshed circuits of interest. The recipes are for the evaluation of inductance matrix entries to a specified accuracy, it is not new quadrature rules for hexahedral domains. This is presented in [9].

5. **Merge groups for additional MLACA-SVD storage reduction:** Within the octree grouping employed by FastHenry, group merging schemes are developed that further reduce the MLACA-SVD memory requirement for a given compression tolerance value. This is presented in [10].

1.3 List of appended papers

The details of this research is reported in two conference papers, included as Appendices A and B [15, 16] and two journal papers, included as Appendices C and D [9, 10]. The conference papers present preliminary results. The main body of work with final results is presented in the journal papers. The appended papers are:

1.3.1 Conference papers

1. **Appendix A: Adaptive cross approximation (ACA) acceleration of superconducting circuit analysis [15]:** The potential benefits of using the ACA as an efficient alternative to the MLFMA in FastHenry is presented in this conference paper. This work links to milestone 1 in Section 1.2. Results show that less

directly-calculated near interactions need to be stored for the ACA. It should be noted that the multilevel fast multipole method (MLFMM) in this paper is referred to as the MLFMA everywhere else in this thesis. These terms are interchangeable. This paper was presented at the International Conference on Electromagnetics in Advanced Applications (ICEAA) in Verona, Italy in September 2017.

2. **Appendix B: Investigation of multilevel adaptive cross approximation (MLACA) acceleration for superconducting circuit analysis [16]:** This conference paper reports on progress with the development of an MLACA solver as an alternative to the existing MLFMA in FastHenry, for compression of the mutual inductance matrix as presented in milestone 2 in Section 1.2. The promising preliminary results show similar memory scaling for MLACA and MLFMA, when using the same near-interaction criterion and integration approach (midpoint integration). This paper was presented at the ICEAA in Cartagena, Colombia in September 2018.

1.3.2 Journal papers

1. **Appendix C: An efficient MLACA-SVD solver for superconducting integrated circuit analysis [9]:** In this journal paper the MLACA-SVD implementation is presented as an alternative to FastHenry's existing MLFMA solver as described in milestones 3 and 4 in Section 1.2. Numerical results for examples of practical interest show that the MLACA-SVD memory scaling versus number of filaments is practically identical to that of FastHenry's MLFMA, and is close to $\mathcal{O}(b \log b)$. This paper shows that the MLACA-SVD requires less memory for the same solution accuracy and that for the examples considered, it is a more efficient solver. This paper is in preparation.

2. **Appendix D: MLACA with modified grouping strategy for efficient superconducting circuit analysis [10]:** In this paper, two modified grouping strategies are proposed to further improve MLACA-SVD efficiency by compressing interactions between larger groups, while maintaining scaling performance consistent with valid admissibility of interactions. This links to milestone 5 in Section 1.2. The strategies are denoted 'shell merging' and 'wall merging'. Results show that required memory is reduced by approximately 30% and that the MLACA-SVD solver with merging requires about four times less memory than FastHenry's MLFMA, for similar accuracy. Shell merging is found to be slightly superior to wall merging. This paper was presented as a poster at the Applied Superconductivity Conference (ASC) in Seattle, USA in November 2018. The paper is currently under review for publication in IEEE Transactions on Applied Superconductivity.

1.4 Thesis structure

The remainder of this thesis is structured as follows:

1. **Chapter 2: Background on FastHenry:** This chapter provides a more thorough background to FastHenry than is provided in the appendices [9, 10, 15, 16].

The formulation of FastHenry's integral equations is adapted from [4] and are also presented in this chapter.

2. **Chapter 3: Implementation of the adaptive cross approximation:** In this chapter, the ACA algorithm background and implementation is discussed in more detail than is provided in the appendices [9, 10, 15, 16]. The ACA is a purely algebraic approach to compression, which emerged after FastHenry was already established. To approximate an asymptotically smooth function, Bebendorf originally presented the ACA [14], [17, 18, 19]. This algorithm has been extended to use SVD for recompression [20, 21]. Based on this work, a new approach to solving superconducting circuits using MLACA-SVD in FastHenry [4], where previously MLFMA [22] had been used, is shown in this chapter.

3. **Chapter 4: Main results:** The main results provided in the journal papers (Appendices C and D) [9, 10] are briefly discussed in this chapter.

4. **Chapter 5: Conclusion:** In this chapter the main conclusion as well as recommendations for further investigations are provided.

5. **Appendices A, B, C and D: Conference and journal papers:** The conference and journal papers which resulted from this work are provided in the appendices.

Chapter 2

Background on FastHenry

This description of FastHenry's formulation of the MQS integral equations is adapted from [4]. The addition of superconducting effects to the FastHenry derivation was not presented in the original release, but was later included. This chapter provides a more thorough background on FastHenry than is provided in the appendices [9, 10, 15, 16].

2.1 Introduction

Inductance extraction of a multi-terminal system is performed by computing the complex frequency dependent impedance matrix under MQS assumptions [4]. The impedance matrix of an h terminal pair problem is denoted $Z(\omega) \in \mathbb{C}^{h\times h}$ with frequency w. An equation to determine $Z(\omega)$ is expressed as

$$Z(\omega)\tilde{I}(\omega) = \tilde{V}(\omega) \tag{2.1}$$

In (2.1), the vectors of terminal current and voltage phasors are $\tilde{I}(\omega), \tilde{V}(\omega) \in \mathbb{C}^h$. Now consider a two-terminal pair shown in Figure 2.1, where the impedance matrix is expressed as

$$Z(\omega) = R(\omega) + j\omega L(\omega) = \begin{bmatrix} R_{11}(\omega) + j\omega L_{11}(\omega) & R_{12}(\omega) + j\omega L_{12}(\omega) \\ R_{21}(\omega) + j\omega L_{21}(\omega) & R_{22}(\omega) + j\omega L_{22}(\omega) \end{bmatrix} \tag{2.2}$$

The resistance matrix is sparse with values only on the diagonal, $R_{12} = R_{21} = 0$. The diagonal resistance matrix entries under superconducting conditions can be purely imaginary, with R_{11} and R_{22} having no real value depending on temperature and material properties. Where the inductance matrix entries L_{11} and L_{22} are the self inductances, the transposed mutual inductances are equal, i.e. $L_{12} = L_{21}$. If the values of vector $\tilde{V}(\omega)$ are known, column i of $Z(\omega)$ can be calculated by taking the row entry i of $\tilde{I}(\omega)$ to be 1 and all the other entries of $\tilde{I}(\omega)$ to be zero. Then column i of $Z(\omega)$ is equal to $\tilde{V}(\omega)$.

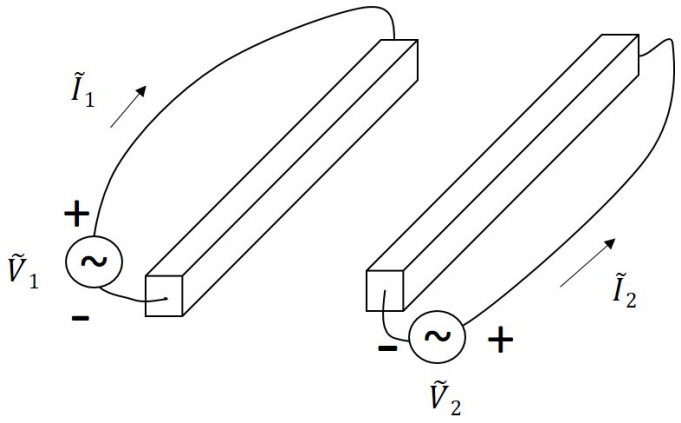

Figure 2.1: Circuit description of a two input-output terminal pairs

2.2 Integral equation formulation

In this section an integral equation to calculate $Z(\omega)$ is formulated. Maxwell's equations in phasor format are shown in (2.3) to (2.6). Under MQS assumptions, displacement current ($j\omega\epsilon\boldsymbol{E}$) is neglected ($\omega$ represents the angular frequency).

$$\nabla \times \boldsymbol{E} = -j\omega\mu\boldsymbol{H}, \tag{2.3}$$

$$\nabla \times \boldsymbol{H} = j\omega\epsilon\boldsymbol{E} + \boldsymbol{J}, \tag{2.4}$$

$$\nabla \cdot (\epsilon\boldsymbol{E}) = \rho, \tag{2.5}$$

$$\nabla \cdot (\mu\boldsymbol{H}) = 0, \tag{2.6}$$

The temperature dependent London penetration depth, $\lambda_L(T)$, can be used to introduce the effect of superconductivity to Maxwell's equations [1]. An equation that can be used to calculate the London penetration depth is shown in (2.7) (T_C is the critical temperature). The London penetration depth characterises the distance that a magnetic field penetrates into the superconductor.

$$\lambda_L(T) = \frac{\lambda_L(0)}{\sqrt{1 - (\frac{T}{T_C})^4}} \tag{2.7}$$

The London penetration depth is used in the superconducting expression for conductivity (2.8). The normal and superconducting channels of the superconductor are taken into account by replacing the conventional conductivity of an ohmic conductor with the following complex conductivity:

$$\sigma = \sigma_C + \frac{1}{j\omega\mu\lambda^2} \tag{2.8}$$

where σ_C is the conductivity of the normal state [1].

In (2.9), the real component is the normal current density; conventional current as a result of normal flowing electrons. The imaginary component is the superconducting current density, which is a result of flowing Cooper pairs. Cooper pairs refer to the superconducting electrons and consist of two electrons that are bound together [1].

Ohm's law is expressed where σ is the complex conductivity. By substituting (2.8) into $\boldsymbol{J} = \sigma \boldsymbol{E}$, an expression for superconducting current density is derived:

$$\boldsymbol{J} = \sigma \boldsymbol{E} = \left(\sigma_C + \frac{1}{j\omega\mu\lambda^2} \right) \boldsymbol{E}. \tag{2.9}$$

For structures that are small compared to the wavelength (low frequencies), the magneto-quasistatic assumption is made that displacement current can be neglected ($j\omega\epsilon\boldsymbol{E} << \boldsymbol{J}$). When the absolute value of complex conductivity (2.8) is increased, the displacement current is further decreased relative to current density (J). Assuming a superconducting state, $\sigma_C = 0$. Considering (2.7), (2.9) and (2.4), increasing $T_C - T$ justifies neglecting displacement current. It follows that the current conservation equation yields $\nabla \cdot \boldsymbol{J} = 0$.

Gauss' Law of magnetic flux is used in the derivation, $\mu\boldsymbol{H} = \nabla \times \boldsymbol{A}$, where $\boldsymbol{A}$ is the magnetic vector potential. Using this with (2.3), results in

$$\nabla \times (\boldsymbol{E} + j\omega\boldsymbol{A}) = 0. \tag{2.10}$$

There is a degree of freedom available when selecting $\boldsymbol{A}$ (gauge invariance), as only its curl is defined. By choosing $\nabla \cdot \boldsymbol{A} = 0$ (Coulomb gauge) and combining this with (2.4) (neglect displacement current), results in $-\nabla^2\boldsymbol{A} = \mu\boldsymbol{J}$. The magnetic vector potential, $\boldsymbol{A}$, can then be expressed as

$$\boldsymbol{A}(\boldsymbol{r}) = \frac{\mu}{4\pi} \int_{V'} \frac{\boldsymbol{J}(\boldsymbol{r}')}{|\boldsymbol{r} - \boldsymbol{r}'|} dv'. \tag{2.11}$$

The permeability is assumed constant $\mu = \mu_0$ [23]. The electric scalar potential can be derived, ϕ that satisfies (2.12).

$$\boldsymbol{E} + j\omega\boldsymbol{A} = -\nabla\phi \tag{2.12}$$

By substituting from Ohm's law, $\boldsymbol{E} = \boldsymbol{J}/\sigma$ and (2.11) into (2.12), the result is the volume integral equation in terms of the current distribution and the electric potential:

$$\frac{\boldsymbol{J}(\boldsymbol{r})}{\sigma} + \frac{j\omega\mu}{4\pi} \int_{V'} \frac{\boldsymbol{J}(\boldsymbol{r}')}{|\boldsymbol{r} - \boldsymbol{r}'|} dv' = -\nabla\phi(\boldsymbol{r}) \tag{2.13}$$

2.2.1 Discretisation and mesh formulation

In the volume integral equation (2.13), the current density is the unknown with infinite degrees of freedom, meaning that a computer cannot calculate this. However, equation (2.13) can be solved in a finite dimentional subspace through the process of applying the method of moments (MoM) [24]. The structure of interest is discretised into b right-angled hexahedral filaments with currents assumed to only flow along the lenghts of these filaments (basis functions direction). This is done to reduce $\boldsymbol{J}(\boldsymbol{r})$ to b degrees of freedom. By choosing a set of basis functions $\boldsymbol{v}_j(\boldsymbol{r}')$, given by

$$\boldsymbol{v}_j(\boldsymbol{r}') = \begin{cases} \dfrac{\hat{\ell}_j}{a_j} & \boldsymbol{r}' \in V_j \\ \\ 0 & \text{elsewhere} \end{cases} \tag{2.14}$$

where $j \in \{1, ..., b\}$, j refers to the source filament numbers, $\hat{\ell}_j$ is the unit vector along the length of the source filament j and a_j is the cross sectional area of source filament j (similarly for i). The j-th filament's domain is defined by V_j. The basis functions can be used to approximate the current density $\boldsymbol{J}(\boldsymbol{r}')$ as

$$\boldsymbol{J}(\boldsymbol{r}') \approx \sum_{j=1}^{b} I_j \boldsymbol{v}_j(\boldsymbol{r}'), \tag{2.15}$$

where I_j is the current along source filament j. This can then be substituted into (2.13), leading to

$$\left(\frac{\hat{\ell}_i}{\sigma a_i}\right) I_i + j\omega \sum_{j=1}^{b} \left(\frac{\mu}{4\pi a_j} \int_{V_j} \frac{\hat{\ell}_j}{|\boldsymbol{r} - \boldsymbol{r}'|} dV'\right) I_j = -\nabla\phi(\boldsymbol{r}) \tag{2.16}$$

Then Galerkin testing is used by selecting the same testing and basis functions. The testing functions are defined as

$$\boldsymbol{w}_i(\boldsymbol{r}) = \begin{cases} \dfrac{\hat{\ell}_i}{a_i} & \boldsymbol{r} \in V_i \\ \\ 0 & \text{elsewhere} \end{cases} \tag{2.17}$$

where $i \in \{1, ..., b\}$; i refers to the testing filament numbers. The testing functions are then multiplied individually with (2.16) and then integrated over the volume. This results in

$$\left(\frac{\ell_i}{\sigma a_i}\right) I_i + j\omega \sum_{j=1}^{b} \left(\frac{\mu}{4\pi a_i a_j} \int_{V_i} \int_{V_j} \frac{\hat{\ell}_i \cdot \hat{\ell}_j}{|\boldsymbol{r} - \boldsymbol{r}'|} dV' dV\right) I_j = \frac{1}{a_i} \int_{a_i} (\phi_{\text{end}} - \phi_{\text{start}}) dA, \tag{2.18}$$

The left of (2.18), where ℓ_i is the length of filament i, can be expressed in matrix form as the impedance matrix ($Z = R + j\omega L$) multiplied with the filament current vector (I_b). The right of this equation, where ϕ_{end} and ϕ_{start} refer to the electric scalar potential at the start and end points of filament i, can then be expressed as the potential difference over all filaments. In matrix form (2.18) is expressed as

$$ZI_b = (R + j\omega L) I_b = V_b. \tag{2.19}$$

The ends of neighbouring filaments can share both nodes (See Figure 2.2), therefore there is no transverse current. No transverse current is modelled, due to the assumption of known direction of flow along long, relatively thin conductors.

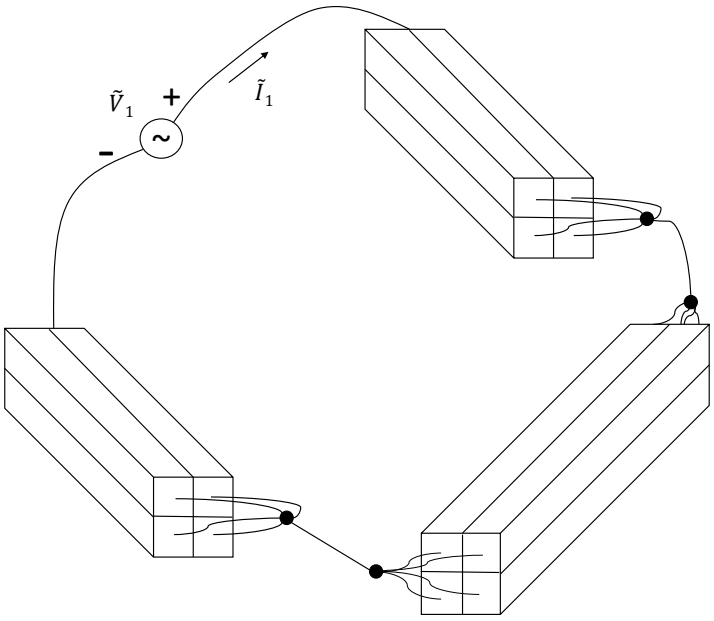

Figure 2.2: A long thin conductor discretised into bundles of filaments sharing nodes. Adapted from [4].

Resistance and inductance matrix entries can be calculated as

$$R_{ii} = \frac{\ell_i}{\sigma(\boldsymbol{r})a_i} \tag{2.20}$$

$$\sigma(\boldsymbol{r}) = \sigma_C(\boldsymbol{r}) + \frac{1}{j\omega\mu\lambda(\boldsymbol{r})^2} \tag{2.21}$$

$$L_{ij} = \frac{\mu}{4\pi a_i a_j}\int_{V_i}\int_{V_j}\frac{\hat{\ell}_i \cdot \hat{\ell}_j}{|\boldsymbol{r}-\boldsymbol{r}'|}dV'dV \tag{2.22}$$

For implementation, the complex component generated by substituting (2.21) into (2.20) is added to the self inductance of (2.22). The PEEC equations for determining resistance and inductance are (2.20) and (2.22) respectively. A mesh-current approach (see Figure 2.3) yields the final system of linear equations [4]

$$MZM^T I_m = V_s \tag{2.23}$$

where m is the total number of linearly independent loops in the mesh and V_s is the (typically sparse) vector of source voltages in all loops, such that $MV_b = V_s$, with M following from Kirchhoff's voltage law applied to all loops. The loop currents relate to the filament currents as $M^T I_m = I_b$.

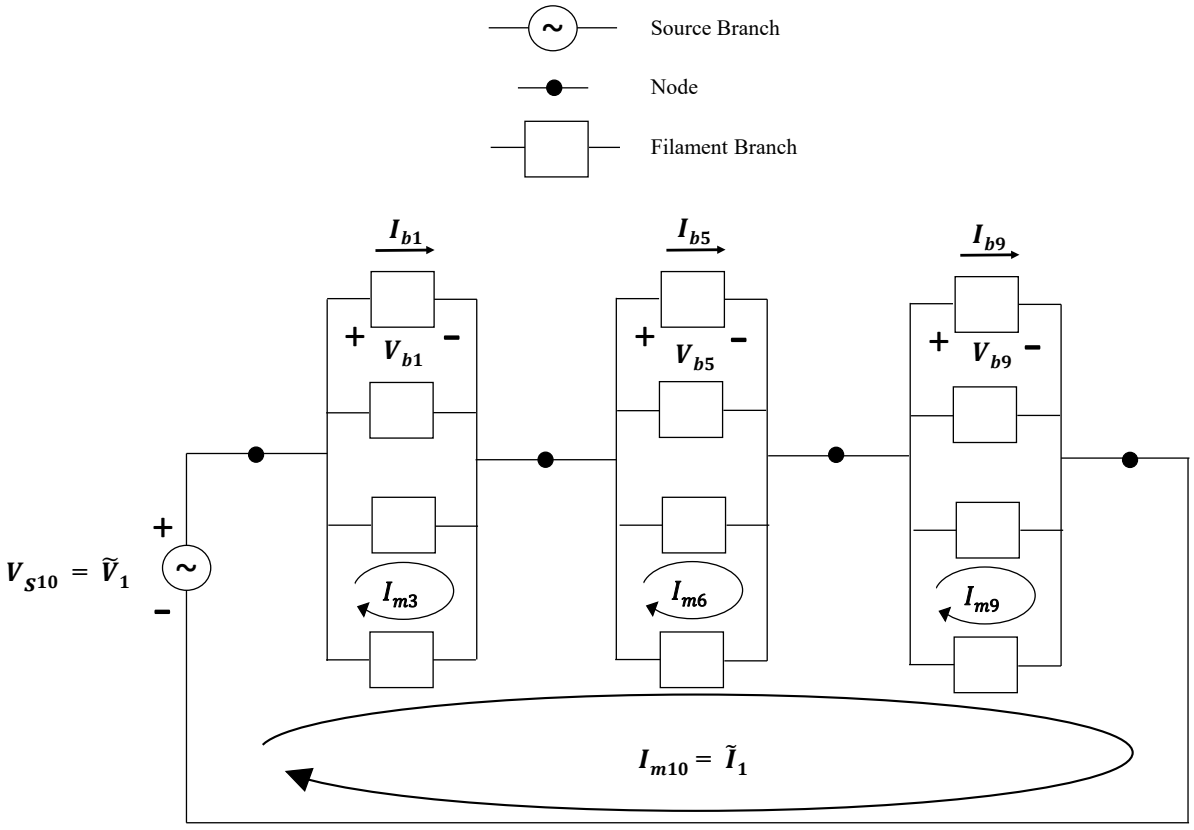

Figure 2.3: A long thin conductor modelled as a circuit. Adapted from [4].

2.3 Multilevel fast multipole algorithm

The MLFMA is a powerful algorithm that can be used for a number of applications. The following relevant aspects of the MLFMA as implemented in FastHenry, will be discussed: how the algorithm computes (2.22) with the multipole expansion; how the multilevel grouping is achieved; which group interaction criteria are used; what expansion order is used and; how this relates back to accuracy, memory and run-time.

A complete description of the FMM is beyond the scope of this thesis, but can be found in Greengard [22]. Nabors and White [7] provide a description of the MLFMA for three-dimensional capacitance extraction (FastHenry's implementation is based on this MLFMA). Kamon and White (FastHenry) [4] provide a description of the MLFMA in the context of inductance extraction as used in this thesis.

To employ the MLFMA, the first step is to define the diagonal, vector-valued basis matrix, with diagonal entries equal to the basis functions evaluated within each element, multiplied with the elemental volume:

$$\boldsymbol{\Lambda}_{ii} = \frac{V_i \hat{\ell}_i}{a_i} = \ell_i \hat{\ell}_i \qquad i \in \{1, ..., b\} \tag{2.24}$$

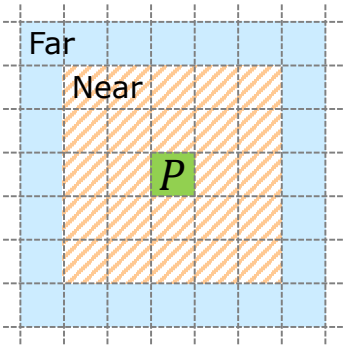

Figure 2.4: Second-nearest neighbours interaction criterion, with respect to group P

Thereafter, the *potential matrix* Φ is defined:

$$\Phi_{ij} = \frac{\mu}{4\pi V_i V_j} \int_{V_i} \int_{V_j} \frac{1}{|\boldsymbol{r} - \boldsymbol{r}'|} dV' dV \qquad i, j \in \{1, ..., b\} \tag{2.25}$$

It follows that the mutual inductance matrix (2.22) can be expressed as:

$$L = \boldsymbol{\Lambda} \cdot \Phi \boldsymbol{\Lambda} \tag{2.26}$$

An MLFMA-representation is established for Φ. The MLFMA-representation rests upon a hierarchical (multilevel), octree grouping of the mesh filaments. Non-self interactions between groups relating to off-diagonal blocks in Φ, are represented in factorised form by exploiting a truncated, series expansion representation of the Green's function $1/|\boldsymbol{r} - \boldsymbol{r}'|$ [7]. Factorisation involves aggregation, translation and disaggregation factors. Aggregation and disaggregation factors can be utilised at multiple levels and the objective is to treat interactions at the highest possible level at which they qualify as far interactions, according to a near-interaction criterion. This criterion is necessary to ensure accuracy of the truncated series expansion. A cubic, second-nearest neighbour criterion is used, as illustrated in Figure 3.1. This means that interactions between groups at leaf level (the smallest group size) are stored directly in case both fall within a $3 \times 3 \times 3$ leaf-level group sized cube.

The accuracy with which this approach represents the true mutual inductance matrix is evaluated in [9] (Appendix C). The near-interaction entries are directly calculated using approximate analytical expressions or quadrature, as automatically determined to be most appropriate. Errors are introduced by the MLFMA into the far-interaction entries, due to the factorised Green's function representation itself, as well as due to the factorisation being done for a midpoint-quadrature based, approximate representation of Φ_{ij}, i.e.

$$\Phi_{ij} \approx \frac{\mu}{4\pi} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} \tag{2.27}$$

where $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ denote the centroids of volumes V_i and V_j, respectively. The reason for this is that the MLFMA implementation is most efficient when dealing with interactions between point sources (i.e. the midpoints), rather than interactions between distributed sources.

As has been shown in [9], midpoint integration is not accurate for approximating the interaction between elongated filaments, unless they are extremely far apart.

Consider a sphere containing a set of filaments. Then apply the multipole expansion to the sphere by only considering the midpoints of the filaments. The FMM algorithm introduces an error in approximating the midpoint integral between any filament inside the sphere with a filament outside the sphere , which is shown in [7] to be

$$\delta_{\text{FMM}} < K\left(\frac{r}{d}\right)^{l+1} \tag{2.28}$$

where δ_{FMM} is the truncated multipole expansion error defined as the absolute value of the exact midpoint integration value minus the approximation. Additionally, r is the sphere radius and d the distance between the centre of the sphere and the interacting filament midpoint. The expansion order denoted l can be used to reduce the error, but at the cost of additional computations. FastHenry uses a default expansion order of 2. In (2.28), K is independent of l, r and d.

2.4 Preconditioned iterative solver

The solver used in FastHenry is a preconditioned, GMRES-based iterative solver. This requires evaluation of the matrix-vector product, which is dealt with in Section 3.6.

In (2.23), the mesh current is solved with GMRES, where I_m is the vector of unknowns. If the inverse of MZM^T is easily accessible, the solution would be trivial. Unfortunately, even if the whole of MZM^T is accessible, computing $(MZM^T)^{-1}$ is computationally expensive. Additionally, when MLFMA or MLACA-SVD is used to approximate most of Z, explicitly calculating MZM^T would negate the storage and run-time benefits of both algorithms.

To circumvent storing the whole MZM^T, the GMRES is used to iteratively minimise the Euclidean norm of the residual, $r^n = MZM^T I_m^n - V_s$ over a Krylov subspace, where n is the iteration number.

This preconditioner is obtained by approximating $(MZM^T)^{-1}$. This approximation can be calculated in a reasonable time by using incomplete LU (ILU) decomposition [25] only considering diagonal entries of the inductance matrix, denoted L_{spar}. The preconditioner generated from $ML_{\text{spar}}M^T$ with ILU is then used in constructing $I_m = (ML_{\text{spar}}M^T)^{-1}x$. Solving for x in the GMRES significantly reduces the number of iterations.

For a thorough description of the GMRES, refer to [6].

Chapter 3

Implementation of the adaptive cross approximation

In this chapter, the ACA algorithm background and implementation is discussed in more detail than is provided in the appendices [9, 10, 15, 16].

The ACA is a purely algebraic approach to compression, which emerged after FastHenry was already established. To approximate an asymptotically smooth function, Bebendorf originally presented the ACA [14], [17, 18, 19]. This algorithm has been extended to use SVD for recompression [20, 21]. Based on this work, a new approach to solving superconducting circuits using MLACA-SVD in FastHenry [4], where previously MLFMA [22] had been used, is shown in this chapter.

3.1 ACA-SVD compression of a mutual inductance, off-diagonal sub-matrix

Consider a sub-matrix denoted by L^{sub}, of the mutual inductance matrix (2.22), representing the interaction between two disjoint groups of filaments, denoted as groups P (consisting of p testing basis functions) and Q (consisting of q source basis functions). It follows that

$$L^{\text{sub}} = \boldsymbol{\Lambda}_P \cdot \Phi_{PQ} \boldsymbol{\Lambda}_Q, \tag{3.1}$$

where Φ_{PQ} represents the appropriate sub-matrix of the global potential matrix (2.25); and $\boldsymbol{\Lambda}_P$ and $\boldsymbol{\Lambda}_Q$ are appropriate, diagonal sub-matrices of the global basis matrix (2.24).

Generally, the ACA is expected to perform well for inter-group matrices of which the entries are proportional to Green's functions associated with various physical boundary value problems [26]; particularly so for the asymptotically smooth, scalar Green's function of Poisson's equation, as featured in (2.22). Performance improves as the distance between groups grow, relative to the group diameters [14, 27, 20]. Scalar testing/source functions present no difficulties, as they effectively just scale matrix rows and columns, which can be normalised. However, irregular scaling of matrix entries within rows and columns can cause catastrophic algorithm breakdown [28, 29]. Such irregular scaling is encountered in L^{sub}, due to the dot-product between the axial unit-vectors. E.g. suppose in the

two groups each filament is orientated in one of two orthogonal directions, then with appropriate numbering of degrees of freedom, L^{sub} takes the form

$$L^{\text{sub}} = \begin{bmatrix} L_{11}^{\text{sub}} & 0 \\ 0 & L_{22}^{\text{sub}} \end{bmatrix}. \tag{3.2}$$

If the ACA algorithm starts by investigating a column intersecting with L_{11}^{sub}, then the algorithm will continue to place crosses centred inside L_{11}^{sub}, until convergence. The part L_{22}^{sub} will never be examined and will falsely be considered to be zero. Therefore, as in the case of the MLFMA compression, the ACA is not applied directly to L^{sub} in (3.1), but rather to Φ_{PQ}. The standard ACA algorithm for this matrix is described in the next section.

3.2 ACA algorithm

The algorithm described in this section is adapted from [19] and based on [14]. Its objective is to obtain the following factorisation:

$$[\Phi_{PQ}]_{p\times q} \approx [U]_{p\times k} \left[V^T\right]_{k\times q}. \tag{3.3}$$

The ACA-estimated rank $k \leq \min\{p, q\}$ (with the expectation that $k \ll \min\{p, q\}$ should hold) is determined by the factorisation error tolerance setting. During the ACA factorisation algorithm, the relative factorisation error, ε_k, is approximately determined at each iteration k, by way of efficient recursive calculations [17]. It is defined and approximated as

$$\begin{aligned} \varepsilon_k &\equiv \frac{\|\Phi_{PQ} - U_k V_k^T\|_F}{\|\Phi_{PQ}\|_F} \\ &\approx \frac{\|U_k V_k^T - U_{k-1} V_{k-1}^T\|_F}{\|U_k V_k^T\|_F} = \frac{\|U_k(:,k)\|_F \|V_k(:,k)\|_F}{\|U_k V_k^T\|_F}, \end{aligned} \tag{3.4}$$

where U_k and V_k denote the factors after k iterations and $\|\cdot\|_F$ represents the Frobenius norm. The algorithm terminates when a specified relative error tolerance level ε_{ACA}, is reached, i.e.

$$\varepsilon_k \leq \varepsilon_{\text{ACA}}. \tag{3.5}$$

To avoid computing the whole of Φ_{PQ}, the ACA algorithm obtains an approximation of this equation by calculating rows and columns of Φ_{PQ}. To initiate the algorithm, a start row to be calculated needs to be selected. To avoid selecting a row outside the bounds of the matrix, the first row is chosen, denoted $\Phi(1,:)$, where the dimentions are $1 \times q$. A row index x is used to keep track of calculated rows and $x_1 = 1$. The first row is then stored in the row error vector used to calculate V denoted δ_V^T, as shown in (3.6).

$$\delta_V^T = \Phi(x_1, :) \tag{3.6}$$

$$\delta_U = \Phi(:, y_1) \tag{3.7}$$

The first column index is selected from the index of the column of the maximum absolute value of δ_V^T and is denoted y_1. The column error vector used to calculate U, δ_U is then calculated as shown in (3.7) where column dimensions are $p \times 1$.

To calculate the first column of V, δ_V is normalised by its maximum absolute value as shown in (3.8). The first column of U is equal to δ_U (3.9). At this point in the algorithm, UV^T would perfectly reconstruct the first row and y_1 column of Φ_{PQ}.

$$V(:,1) = \frac{\delta_V}{|\,[\delta_V]_{y_1}\,|} \tag{3.8}$$

$$U(:,1) = \delta_U \tag{3.9}$$

The norm $||\Phi_{PQ}||_F$ is approximated recursively and updated after every iteration. The contribution from the first iteration ($\|\Phi^{(1)}\|_F$) is calculated as shown in (3.10).

$$\|\Phi^{(1)}\|_F^2 = \|U(:,1)\|_F^2 \|V(:,1)\|_F^2 \tag{3.10}$$

The first iteration of the algorithm (3.6−3.10) is followed by a general case for the k^{th} iteration. The k^{th} iteration starts by calculating the row index x_k as the index of the maximum absolute value of δ_U from a row not previously calculated. This is then followed by calculating δ_V as shown in (3.11). Next, the column index, y_k is calculated as the index of the maximum absolute value of δ_V^T from a column not previously calculated. This allows for δ_U to be calculated as shown in (3.12).

$$\delta_V^T = \Phi(x_k,:) - \sum_{c=1}^{k-1} U(x_k,c)V(:,c)^T \tag{3.11}$$

$$\delta_U = \Phi(:,y_k) - \sum_{c=1}^{k-1} U(:,c)V(y_k,c) \tag{3.12}$$

$$V(:,k) = \frac{\delta_V}{|\,[\delta_V]_{y_k}\,|} \tag{3.13}$$

$$U(:,k) = \delta_U \tag{3.14}$$

Then V_k is updated as shown in (3.13) and U_k is updated as shown in (3.14). The recursive update of $\Phi^{(k)}$ is shown in (3.15).

$$\|\Phi^{(k)}\|_F^2 = \|\Phi^{(k-1)}\|_F^2 + \|U(:,k)\|_F^2 \|V(:,k)\|_F^2 + 2\sum_{c=1}^{k-1} |U(:,c)^T U(:,k)| \cdot |V(:,c)^T V(:,k)| \tag{3.15}$$

The algorithm terminates if condition (3.16) is satisfied. This condition is the approximation shown by the substitution of (3.4) into (3.5).

$$\|U(:,k)\|_F \|V(:,k)\|_F \leq \varepsilon_{\text{ACA}} \|\Phi^{(k)}\|_F \tag{3.16}$$

The approximation's storage requirement is $\mathcal{O}(k(p+q))$ rather than $\mathcal{O}(p \times q)$ required to store Φ_{PQ} directly.

3.3 SVD recompression

Generally, the columns of neither U nor V are orthogonal [21]. Therefore, singular value decomposition (SVD) is further applied to eliminate any possible redundancy within U and V, as recommended in [20, 21]. First, QR decompositions of U and V are computed using the Householder algorithm [30], as

$$U = [Q_U]_{p\times k}\,[R_U]_{k\times k} \tag{3.17}$$
$$V = [Q_V]_{q\times k}\,[R_V]_{k\times k}\,. \tag{3.18}$$

Then apply SVD to $R_U R_V^T$:

$$R_U R_V^T = \tilde{U}\Sigma\tilde{V}^T. \tag{3.19}$$

This yields the desired SVD of the product UV^T:

$$UV^T = Q_U\tilde{U}\Sigma\left(Q_V\tilde{V}\right)^T. \tag{3.20}$$

The singular values are found in descending order on the diagonal of Σ, with the largest denoted by $\sigma_{\max}$. All values falling below a relative threshold are removed [20]. The threshold is defined as

$$\frac{\sigma_i}{\sigma_{\max}} \leq \varepsilon_{\text{SVD}} \qquad i \in \{1, ..., k\}. \tag{3.21}$$

This results in a new rank $\tilde{k} \leq k$ and recompression of the original U and V matrices. Since the SVD rank is precise, while the ACA rank is approximate, a $10\times$ buffer factor is employed to ensure reliable ACA-SVD accuracy:

$$\varepsilon_{\text{SVD}} = 10\varepsilon_{\text{ACA}}. \tag{3.22}$$

The SVD tolerance is then considered the actual matrix compression tolerance.

The additional computational cost of adding SVD on top of ACA does not change the overall cost scaling of the ACA. The runtime for steps (3.17), (3.18) and (3.20) together, scales as $\mathcal{O}(k^2(p+q+k))$ [30].

3.4 ACA requirements

The two main conditions required for ACA are; that the integration kernel be asymptotically smooth and; that the interacting groups' diameters be small enough relative to the distance between them. If these conditions are not met, reliable ACA results cannot be guarenteed. These conditions are discussed in this section.

The 3D Green's function is

$$G(\boldsymbol{r}, \boldsymbol{r}') = \frac{1}{4\pi|\boldsymbol{r} - \boldsymbol{r}'|}. \tag{3.23}$$

Although it is widely known that this kernel is asymptotically smooth everywhere except for on the diagonal [20], where $\boldsymbol{r} = \boldsymbol{r}'$, for completeness the process of determining asymptotic smoothness is described [31, 28, 18]: Consider $\boldsymbol{r}_i$ and $\boldsymbol{r'}_j$ being two point systems

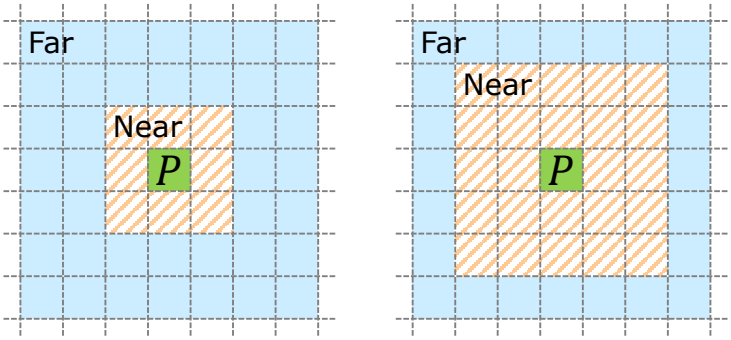

Figure 3.1: Two near-interaction criteria, with respect to group P. Left: nearest neighbours. Right: second-nearest neighbours.

(grids) in three-dimensional sets defined in (3.24) and (3.25). These grids are occupied by points on right-angled hexahedral filaments discussed in Subsection 2.2.1.

$$\boldsymbol{r}_i = (x_i, y_i, z_i) \tag{3.24}$$

$$\boldsymbol{r'}_j = (x_j, y_j, z_j) \tag{3.25}$$

As G can be rewritten as $G(\boldsymbol{r}, \boldsymbol{r'}) = G(\boldsymbol{s})$, where $\boldsymbol{s} = \boldsymbol{r} - \boldsymbol{r'}$ and if $\boldsymbol{s} \neq 0$, G is differentiable everywhere.

The criteria for asymptotic smoothness is stated as

$$|D^\alpha G(\boldsymbol{s})| \leq |\alpha|! C_1 C_2^{|\alpha|} \|\boldsymbol{s}\|^{g-|\alpha|}, \tag{3.26}$$

where the multi-index $\alpha \in \mathbb{N}_0^3$ and $|\alpha| = \alpha_1 + \alpha_2 + \alpha_3$; the Euclidean norm is denoted by $\|\cdot\|$; constants C_1 and C_2 are both larger than zero; $g \leq 0$; C_1, C_2 and g are independent of $|\alpha|$ and; the mixed partial derivatives are D^α as shown by

$$D^\alpha = \frac{\partial^{\alpha_1} \partial^{\alpha_2} \partial^{\alpha_3}}{(\partial x)^{\alpha_1} (\partial y)^{\alpha_2} (\partial z)^{\alpha_3}}. \tag{3.27}$$

Next, consider the η-admissibility condition. For the ACA on a given level of the octree, the η-admissibility condition, which predicts exponential decay of singular values for asymptotically smooth kernels [14, 27], can be stated as follows:

$$\operatorname{diam}(D_Q) \leq \eta \operatorname{dist}(D_P, D_Q) \qquad \{0 < \eta\}, \tag{3.28}$$

where D_P and D_Q are taken as the octree cubes (identical in size) defining the testing and source filament groups, respectively. If there is exponential decay of singular values, k is logarithmically dependent on $1/\varepsilon_{\text{ACA}}$. Increasing the distance between D_P and D_Q results in more rapid singular value decay, therefore reducing η is regarded as strengthening the admissibility condition. Figure 3.1 illustrates two near-interaction criteria, depicted in two dimensions for simplicity. FastHenry's MLFMA uses the second-nearest neighbour criterion exclusively. The MLACA-SVD is tested with this same criterion. It corresponds to $\eta < \sqrt{3}/2$, which is a strong admissibility condition. The MLACA-SVD is also tested

with the nearest-neighbour criterion. It is a weaker admissibility condition, with $\eta < \sqrt{3}$, which should theoretically result in a less accurate ACA approximation for the same approximate rank [20]. The benefit of the nearest-neighbour criterion is that valid far interactions generally occur at a higher level, allowing ACA-SVD compression of larger sub-matrices. For an evaluation of the relative efficiencies of these criteria as well as for more details on the implementation of MLACA-SVD inside FastHenry, see Appendix C [9].

3.5 Integration techniques

In determining mutual inductance matrix entries with MLACA-SVD, a quadrature recipe (see Appendix C) [9] is used, in contrast to FastHenry's MLFMA that uses midpoint-integration. This ensures the rigorous evaluation of all matrix entries to a higher accuracy than the specified ACA error tolerance, such that accuracy is fully controlled.

Rules on when to use analytic and when to use Gaussian (Gaussian-Legendre) quadrature to determine inductance values are presented in Appendix C [9]. For the inner (source) integral in the integral equation (2.22), either Gaussian quadrature [32] in product-rule format is used, or analytical evaluation [33]. The choice depends on the distance to the testing domain, since Gaussian quadrature requires a smooth integrand. As the testing and source domains get closer, the near-singular (less smooth) behaviour of the kernel increases. Gaussian quadrature is always used for the outer integrand, as it is sufficiently smooth.

The rest of this section provides a discussion on the implementation of analytic integration used on the inner integral for right-angled hexahedral filaments.

Analytic integration is used on the inner integral (V_j) of the integral equation (2.22) when calculating self-inductance or the mutual inductance of nearby filaments. A full description of the process used to perform analytic integration described for polyhedrons as a source distribution is presented in [33]. As the polyhedrons of interest are right-angled hexahedrals, a specific implementation is derived. The process entails integrating over the source by considering all six faces by navigating over their edges. The integration path along the edges is taken as right-handed with respect to the norm vector ($\hat{\boldsymbol{n}}$) directed outwards from the face as seen in Figure 3.2. The integration path is along the unit vector $\hat{\boldsymbol{l}}$ from the edge end ρ^- to ρ^+.

The equation used to sum over the $i = 4$ edges of the $j = 6$ faces, to obtain the total contribution of the source filament from the perspective of an observation point is shown in (3.37), where d_j is the projection displacement from the observation onto the plane (positive direction $\hat{\boldsymbol{n}}_j$); $\hat{\boldsymbol{u}}_{ij}$ is the unit vector on the plane perpendicular to the edge and away from the face calculated as (3.30) and; P^0_{ij} is the unit vector in the direction from the observation projection onto the plane to the extended edge perpendicularly. All remaining variables can be read off Figure 3.2 and calculated using equations (3.29−3.36) where ρ_j is the vector from (O'), the origin's perpendicular projection onto the plane to the perpendicular projection of the observation point onto the plane.

$$\hat{\boldsymbol{l}}_{ij} = \frac{\rho_{ij}^{+} - \rho_{ij}^{-}}{|\rho_{ij}^{+} - \rho_{ij}^{-}|} \tag{3.29}$$

$$\hat{\boldsymbol{u}}_{ij} = \hat{\boldsymbol{l}}_{ij} \times \hat{\boldsymbol{n}}_{j} \tag{3.30}$$

$$l_{ij}^{\pm} = (\rho_{ij}^{\pm} - \rho_{j}) \cdot \hat{\boldsymbol{l}}_{ij} \tag{3.31}$$

$$P_{ij}^{0} = |(\rho_{ij}^{\pm} - \rho_{j}) \cdot \hat{\boldsymbol{u}}_{ij}| \tag{3.32}$$

$$\hat{\boldsymbol{P}}_{ij}^{0} = \frac{(\rho_{ij}^{\pm} - \rho_{j}) - l_{ij}^{\pm}\hat{\boldsymbol{l}}_{ij}}{P_{ij}^{0}} \tag{3.33}$$

$$P_{ij}^{\pm} = |\rho_{ij}^{\pm} - \rho_{j}| \tag{3.34}$$

$$R_{ij}^{O} = \sqrt{(P_{ij}^{0})^2 + d_{j}^2} \tag{3.35}$$

$$R_{ij}^{\pm} = \sqrt{(P_{ij}^{\pm})^2 + d_{j}^2} \tag{3.36}$$

$$\begin{aligned}\int_{V} \frac{dV'}{|\boldsymbol{r} - \boldsymbol{r}'|} = \frac{1}{2}\sum_{j}^{6} d_{j}\Bigg\{ \sum_{i}^{4} \hat{\boldsymbol{P}}_{ij}^{0} \cdot \hat{\boldsymbol{u}}_{ij} \times \Bigg[|d_{j}| \Bigg(\arctan \frac{P_{ij}^{0} l_{ij}^{+}}{(R_{ij}^{0})^2 + |d_{j}| R_{ij}^{+}} \\ - \arctan \frac{P_{ij}^{0} l_{ij}^{-}}{(R_{ij}^{0})^2 + |d_{j}| R_{ij}^{-}} \Bigg) - P_{ij}^{0} \ln \frac{R_{ij}^{+} + l_{ij}^{+}}{R_{ij}^{-} + l_{ij}^{-}} \Bigg] \Bigg\}\end{aligned} \tag{3.37}$$

Details of the geometric parameters are shown in Figure 3.2, where O is the coordinate origin, position vector $\boldsymbol{r}$ refers to the observation point and C is the line segment (edge). A complete description of (3.29–3.37) and Figure 3.2 can be found in [33].

The first step in implementing analytic integration, is to project the origin (O) perpendicularly onto the infinite plane ($\mathcal{P}$) created by the face of interest. This projection is determined by using the equation for a line in three-dimensional space (3.38) with gradient in the vector direction (a, b, c) orthogonal to the plane of interest ($\mathcal{P}$) and passing through the origin at (x_0, y_0, z_0). The equation for a plane (3.39) is constructed by taking a point on the face (x_1, y_1, z_1) and vector direction orthogonal to the plane (a, b, c), therefore equivalent to the gradient required for the line. A similar approach is used to obtain the projection of the observation point onto the plane by solving (3.38) and (3.39) simultaneously. A step-wise process follows to determine the remaining required geometrical quantities with the assistance of (3.29–3.36), as can be seen in Figure 3.2. This process is repeated for all twenty-four edges. Note that in (3.38), if $a = 0$ then $x = x_0$ and the $((x - x_0)/a)$ part of the equation is dropped, similarly for $b = 0$ and $c = 0$.

$$\frac{x - x_0}{a} = \frac{y - y_0}{b} = \frac{z - z_0}{c} \tag{3.38}$$

$$0 = a(x - x_1) + b(y - y_1) + c(z - z_1) \tag{3.39}$$

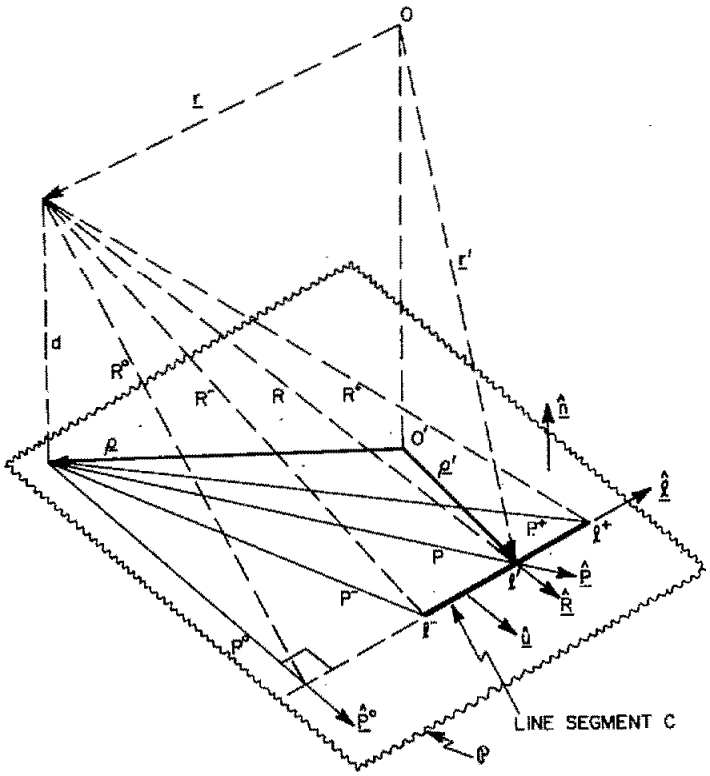

Figure 3.2: Geometrical quantities on the plane $\mathcal{P}$ for line segment C. The coordinate origin is O and at position vector $\boldsymbol{r}$ [33]

3.6 Matrix-vector product

Exactly the same preconditioner and GMRES iteration scheme used for the MLFMA solver is used for the MLACA-SVD solver (as noted in Section 2.4).

$$MZM^T I_m^n = MRM^T I_m^n + j\omega MLM^T I_m^n \tag{3.40}$$

As stated in [4], after splitting Z into its real and imaginary parts (3.40), $MRM^T I_m^n$ can be calculated in $\mathcal{O}(m)$ operations as R is diagonal and the sparse mesh matrix M has $\mathcal{O}(m)$ nonzero terms. Theoretically, a multilevel implementation computing $MLM^T I_m^n$ with the near interactions of L, also require $\mathcal{O}(m)$ operations. Note, as previously stated for FastHenry's superconducting implementation, the imaginary part of R is stored on the diagonal of L and also note that I_m^n is complex.

For each iteration, $M^T I_m^n$ is determined and temporarily stored. The preconditioner of Section 2.4 is used for I_m^n. Additionally, each iteration $\ell_i\hat{\ell}_i M^T I_m^n$ is computed and temporarily stored three times to account for all three principle component directions. Computing $\ell_i\hat{\ell}_i M^T I_m^n$ does not change computational scaling. The rest of this section discusses the number of computations required for the MLACA-SVD compressed matrix-vector product of an off-diagonal block. From here on $w = \ell_i\hat{\ell}_i M^T I_m^n$.

In the matrix-vector product, $UV^T w$ is computed. By splitting the matrix-vector product into $V^T w$ and $U(V^T w)$, the number of extra additions and multiplications can

be calculated. The first multiplication $V^T w$ requires $k \times q$ multiplications and $k(q-1)$ additions. The second multiplication $U(V^T w)$ requires $k \times p$ multiplications and $p(k-1)$ additions. Summing this, results in $2k(p+q) - p - k$ extra multiplications and additions from the two multiplications. As this process needs to be done for all three principle directions, the total changes to $3(2k(p+q) - p - k)$.

For each iteration, $\ell_i \hat{\ell}_i$ $(\boldsymbol{\Lambda})$ is again multiplied out three times and M is multiplied out at the end of the iteration. This once again does not affect computational scaling. Note that the number of overall operations for the matrix-vector product is effectively proportional to the storage of the MLACA-SVD factors.

3.7 Group merging

This section describes new strategies to merge groups on a given ACA level, such that larger blocks of Φ are factorised at a time for higher compression efficiency. These strategies were developed and reported in Appendix D [10].

As can be deduced from (3.3), provided that the approximate rank k is negligibly affected by increasing group filament numbers p and q, then increasing p and q results in a greater compression percentage (storage reduction). As an example, applying ACA to two dense neighbouring groups together, containing all filaments, is not appropriate as there will likely be no exponential decay of singular values ($\eta-$admissibility condition). Therefore a more subtle approach is required.

With ACA, the nearest neighbours' interaction criteria (Figure 3.1) can be used to handle large groups at high levels, as stated in section 3.4. Maintaining the octree grouping of FastHenry, an interaction grouping scheme to increase the number of filaments in interacting groups is presented in this section. A further discussion on the two approaches to group merging can be found in Appendix D [10].

When performing group merging, group diameters are no longer necessarily equal. Therefore the $\eta-$admissibility condition of (3.28) is provided in a more general form, as (3.41). The alternative non-convex approach shown in Figure 3.3 is used to find parameters for the $\eta-$admissibility condition. The merging approaches that follow lend themselves to restricting both observation and non-convex source group diameters. Therefore the strong $\eta-$admissibility condition (3.41) can be used. For a thorough description of the admissibility condition, applicable to (3.41), refer to [27].

$$\max\{\mathrm{diam}(D_P), \mathrm{diam}(D_Q)\} \leq \eta \ \mathrm{dist}(D_P, D_Q), \qquad \{0 < \eta\} \tag{3.41}$$

Two approaches to group merging were developed. These are shell merging and wall merging. In both of these merging strategies, each group on a given octree level is in turn considered as observer to compress at least all of its far interactions with source groups, before moving on to the next one. The last observer is not considered since by definition its interactions would have already been dealt with. For merging purposes, the observer group under current consideration is labelled the anchor group.

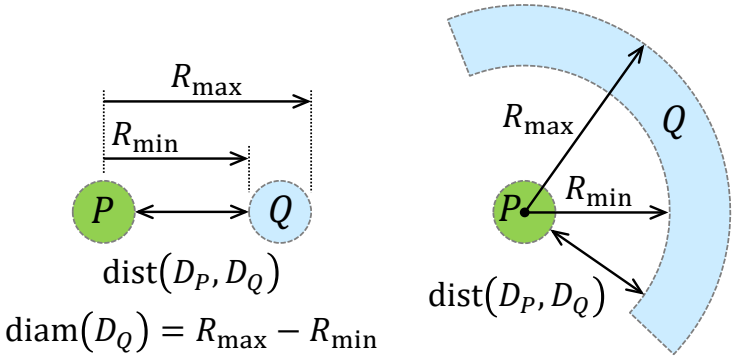

Figure 3.3: Left: alternative method of group diameter measurement. Right: set up with a non-convex source group, for which the alternative η-admissibility condition is identical to that of the set up on the left.

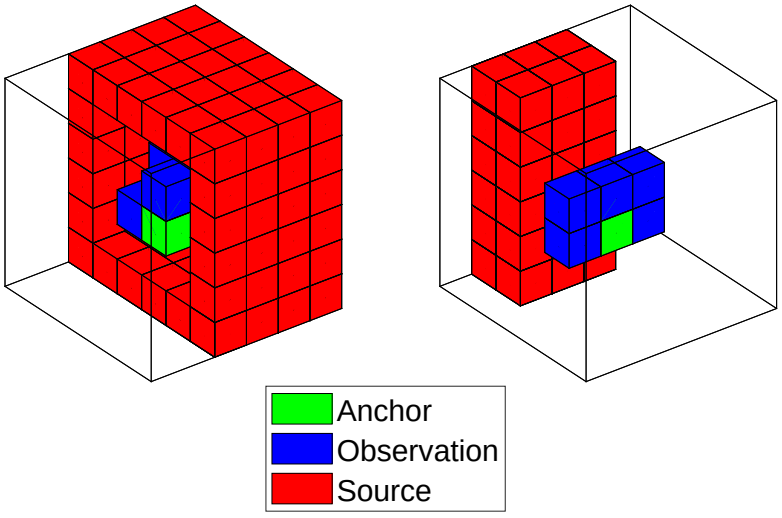

Figure 3.4: Examples of the two group merging strategies: shell merging (left) and wall merging (right). The coloured cubes represent octree groups and the large wire-frame cube depicts the bounding box of all of the anchor's far-interaction source groups. Note that in the figure on the left (shell merging), some of the shared-interaction source groups are not shown for clarity (i.e. one third of the merged "shell" is cut away).

Shell merging aims to merge a compact core of observer groups, for which a 'shell' of valid source groups are identified and merged. Wall merging on the other hand, aims to construct parallel 'walls' of merged observation and source groups. Figure 3.4 shows examples of the two strategies. See Appendix D [10] for the full details.

Chapter 4

Main results

In this chapter the main results from the journal papers [9, 10] are presented and discussed. Firstly the MLACA-SVD with integration rules (Appendix C) is presented. This is followed by results with group merging added (Appendix D). All ACA related results presented in this chapter are obtained with integration recipe $\varepsilon_{\text{quad}} < 10^{-6}$ as discussed in Appendix C.

4.1 Main results from Appendix C: Efficient MLACA-SVD solver

After promising preliminary memory scaling results shown in the conference papers [15, 16], the first set of results indicating significant improvements to FastHenry are shown in Appendix C and briefly discussed in this section.

Results in Appendix C show improved accuracy with the implementation of MLACA-SVD when compared to FastHenry's MLFMA implementation. Considering that elongated filaments are generally used in meshes and that midpoint integration often provides a poor approximation of these filaments' interactions, using an alternative integration approach is desired for interactions to be compressed. Figure 4.1 shows the superconducting, integrated circuit test model used to evaluate the performance of the MLACA-SVD solver. This is an energy-efficient single flux quantum (eSFQ) circuit [34]. Performance with regards to memory requirement and accuracy with which the true matrix L (2.22) is approximated by the compressed versions, is considered. The label 'MLFMA' refers to results obtained with FastHenry's MLFMA solver with default settings that employs the standard FastHenry routines to evaluate the near-interaction matrix entries.

4.1.1 Memory

The total memory required to store the mutual inductance matrix L (2.22) in compressed form, is measured as the mesh density is varied. Meshing is done with the InductEx package [8]. For the MLACA-SVD solver the near-interaction criteria of Figure 3.1 are both considered, as well as four factorisation tolerance settings $\varepsilon_{\text{SVD}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Figure 4.2 shows the memory scaling results for the eSFQ model.

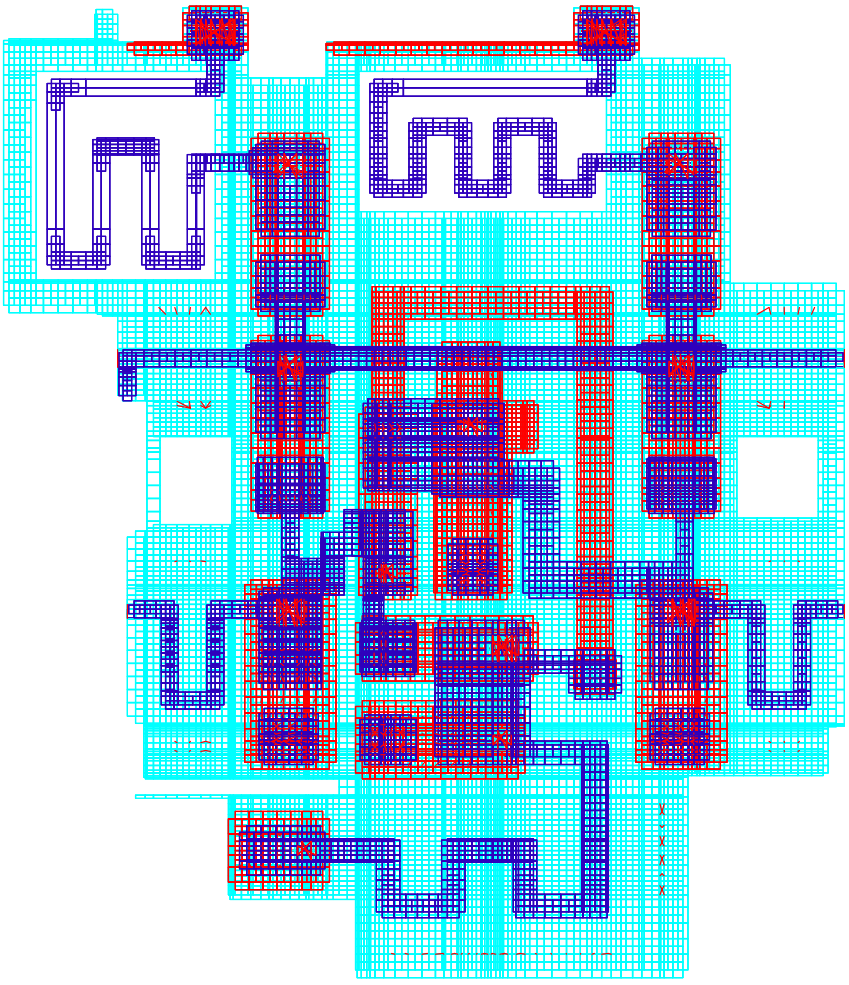

Figure 4.1: Example mesh of a test model representing a superconducting, integrated eSFQ circuit structure.

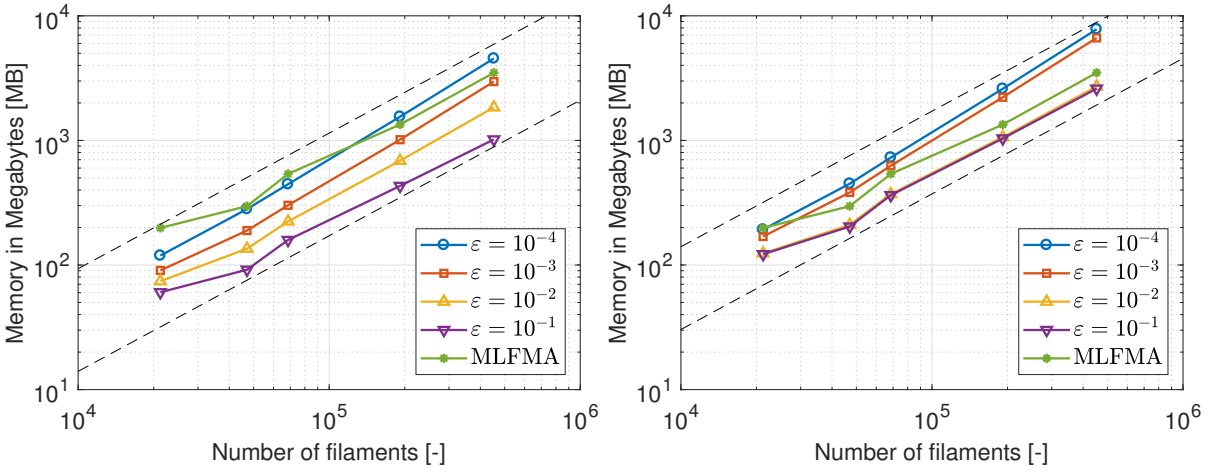

Figure 4.2: Compressed matrix memory requirement vs. number of filaments, for the eSFQ circuit. The total number of filaments is varied from $21,251$ to $452,874$; tolerance values refer to ε_{SVD}. Left: MLACA-SVD with nearest-neighbour criterion. Right: MLACA-SVD with second-nearest neighbour criterion.

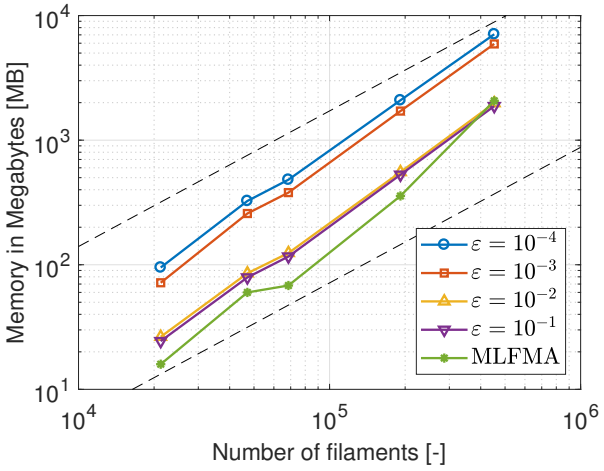

Figure 4.3: Compressed matrix memory requirement with near-interaction storage excluded, for MLACA-SVD and MLFMA with the same, second-nearest neighbour criterion for the eSFQ circuit. Tolerance values refer to ε_{SVD} and $\mathcal{O}(b \log b)$ trend lines are shown.

Firstly, consider the observed scaling orders. Dashed trend lines of $\mathcal{O}(b \log b)$ show that the MLACA-SVD versions scale at this rate or slightly above it. This is expected for static/low-frequency solutions with asymptomatically smooth kernels [19]. According to [4], the expected scaling of the MLFMA is $\mathcal{O}(b)$ and here it does seem to scale on average, a little better than the MLACA-SVD versions. However, note that the MLFMA does not exploit symmetry in the storage of its directly-calculated, near-interaction entries. At low mesh sizes these tend to dominate the MLFMA memory requirement. To remove this influence, the MLFMA is compared with the second-nearest neighbour criterion MLACA-SVD, with near-interaction storage excluded for both, i.e. purely the compressed storage of exactly the same sets of matrix entries are compared. Figure 4.3 shows the results, which indicate that for the considered test case, the MLFMA in fact scales the same or even slightly worse than the MLACA-SVD.

Secondly, consider the comparative memory requirements of Figure 4.2 in absolute terms. Clearly, applying the nearest-neighbour criterion MLACA-SVD always yields a lower memory requirement than with the second-nearest criterion version, for the same mesh and SVD tolerance. Also clearly, the MLACA-SVD memory requirement is strongly and uniformly influenced by the choice of SVD tolerance. Arguably the most important observation, is that for the nearest-neighbour criterion, MLACA-SVD out-performs the MLFMA. However, this should be considered in conjunction with the relative accuracies of these compressed representations. Accuracy is considered in the next section.

4.1.2 Accuracy

The test model has a number of defined ports. At each port in turn, a unit voltage source is connected with all others shorted out. After solving the global current distribution, a vector of the ports currents is obtained. These vectors form the columns of a port current matrix I_{port} that is the key input to circuit parameter extraction tools such as InductEx [8].

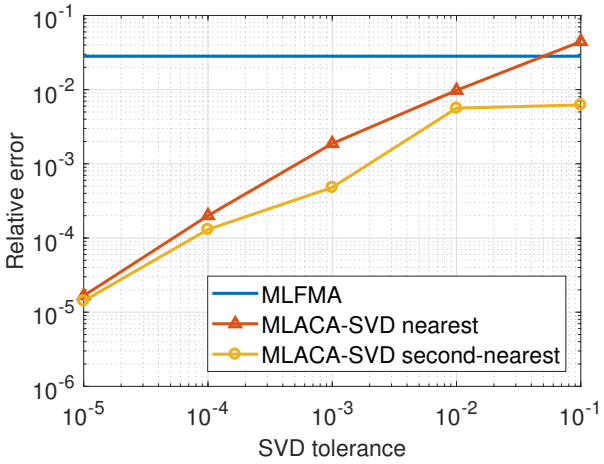

Figure 4.4: Relative port current matrix errors vs. SVD tolerance for the eSFQ circuit model with 38,895 filaments.

Accuracy of the compressed representations is evaluated by way of relative port current matrix errors, defined as

$$\text{Relative error} = \frac{\left\| I_{\text{port}}^{\text{approximate}} - I_{\text{port}}^{\text{reference}} \right\|_F}{\left\| I_{\text{port}}^{\text{reference}} \right\|_F}. \tag{4.1}$$

Reference port current matrices $I_{\text{port}}^{\text{reference}}$ are obtained by calculating all mutual inductance matrix entries with the $\varepsilon_{\text{quad}} < 10^{-6}$ recipe, for an MLACA solution ($\varepsilon_{\text{ACA}} = 10^{-7}$, no SVD recompression) in the case of the eSFQ circuit (38,895 filaments). The latter solution together with the approximate solutions $I_{\text{port}}^{\text{approximate}}$ (with MLACA-SVD or MLFMA) are obtained with the iterative solver convergence criterion set to an extremely small value, such that the error is only determined by the accuracy of the compressed matrix representation (i.e. these are essentially direct solutions with the reconstituted matrices).

Figure 4.4 shows the results. A relative error of $\sim 10^{-2}$ is introduced by the MLFMA. This is due to errors in calculating the near-interaction entries, as well as the coarse, midpoint quadrature upon which the compression is based (see (2.27)) and the approximate nature of the compression itself. On the other hand, the accuracy with which the MLACA-SVD represents the true matrix is fully controllable through parameter ε_{SVD} (3.21).

An important implication is that when using a mesh which is sufficiently fine to yield a very accurate current distribution solution (when rigorously evaluating the matrix entries), the accuracy can be limited by the MLFMA-introduced error. This is not an issue for the MLACA-SVD that can in principle solve the electromagnetic field problem to any desired accuracy.

The results of Figure 4.4 together with Figure 4.2, indicate that the MLACA-SVD is more efficient than FastHenry's MLFMA. Particularly, the nearest-neighbour criterion together with either $\varepsilon_{\text{SVD}} = 10^{-3}$ or $\varepsilon_{\text{SVD}} = 10^{-2}$ is both more accurate than the MLFMA and require less memory than the MLFMA. For the MLACA-SVD, the nearest-neighbour

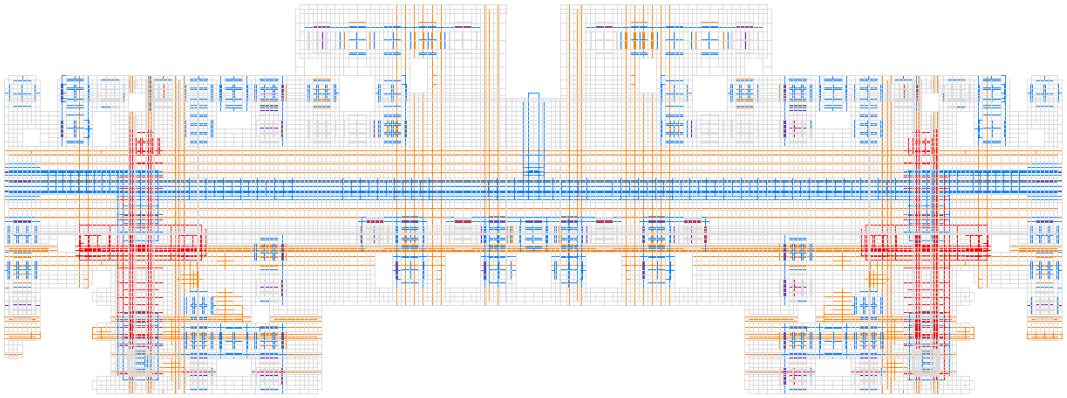

Figure 4.5: Test model representing an inductance test SQUID for experimental measurements, with loop inductance in M6 sandwiched between a ground plane in M4 and a ground-connected sky plane in M7. [36].

criterion is clearly more efficient than the second-nearest neighbour criterion, when considering accuracy vs. memory. Nevertheless, the nominal, improved accuracy obtained with the larger near-interaction criterion for the same mesh, does warrant discussion. It is partly due to the nearer interaction criterion simply resulting in a larger percentage of the matrix being approximated [35] (assuming that near interactions are calculated more accurately than compressed ones). However, it is mainly due to the relationship between the η-admissibility condition (3.28) and accuracy, as explained in Appendix C [9].

4.2 Main results from Appendix D: MLACA with modified grouping

Figure 4.5 shows the superconducting integrated circuit test model [36], which represents an inductance test SQUID. It is meshed with varying filament numbers, to obtain all of the results.

Consider a fixed mesh with $24,272$ filaments. The SVD error tolerance ε_{SVD}, which controls the MLACA-SVD matrix compression error [9], is varied. Figure 4.6 compares the normal MLACA-SVD, shell-merging MLACA-SVD and wall-merging MLACA-SVD, with FastHenry's MLFMA (the latter gives fixed results, because ε_{SVD} is not relevant to it). All with nearest-neighbour criterion have the normal MLACA-SVD's logarithmic dependence of memory requirement upon ε_{SVD} is preserved by the merging versions. Both merging versions yield a fixed reduction in required memory of approximately 30%, with wall merging being slightly superior. A fifth trace is also shown, for normal MLACA-SVD with the self near-interaction criterion of Figure 4.7. This criterion implies $\eta = \infty$ in (3.41), which is not a valid condition. Its memory scaling is dramatically poorer than the others. This result serves to emphasise that the two merging strategies are consistent with maintained admissibility. The solution accuracy results shown in Figure 4.6 are the relative errors in the port currents, as defined in (4.1). These results show that conver-

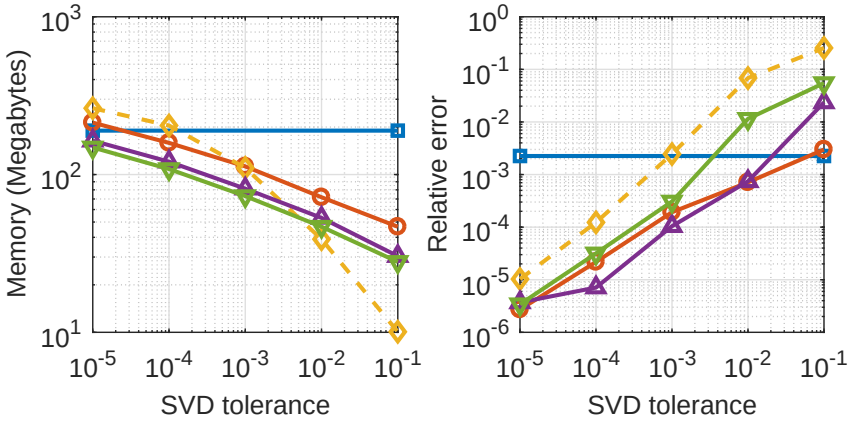

Figure 4.6: Memory required to store the mutual inductance matrix (left) and relative error in the solution port currents (right). Both quantities are shown as functions of the matrix compression error tolerance ε_{SVD}, for a mesh of 24,272 filaments.

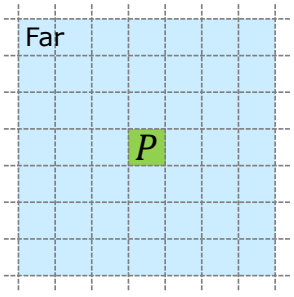

Figure 4.7: Self-interaction criterion, with respect to observer group P.

gence towards the true mutual inductance matrix as compression tolerance is decreased, is fairly similar for the normal MLACA-SVD and the two merging versions, especially for $\varepsilon_{\text{SVD}} \leq 10^{-3}$. At $\varepsilon_{\text{SVD}} = 10^{-2}$ these three schemes along with FastHenry's MLFMA, all yield relative errors within about a factor of 10 from each other. As expected, the normal MLACA-SVD with self criterion fares poorly. Shell merging generally yields superior accuracy to wall merging, which is ascribed to the geometric properties of the former (see Figure 3.4) more closely adhering to the idea of maintained (alternative) admissibility as expressed in Figure 3.3. Taking the results of Figure 4.6 all together, shell merging may be regarded as slightly more efficient than wall merging.

Now consider memory scaling with respect to the number of filaments, as shown in Figure 4.8. The merging versions scale exactly like the normal MLACA-SVD. Furthermore, the reduction by approximately 30% relative to the normal MLACA-SVD is affirmed for both merging versions, with wall merging again being slightly superior when only memory is considered. Note that for $\varepsilon_{\text{SVD}} = 10^{-2}$, where the merging versions yield compression accuracies similar to FastHenry's MLFMA, the memory requirement is less, by about a factor of 4.

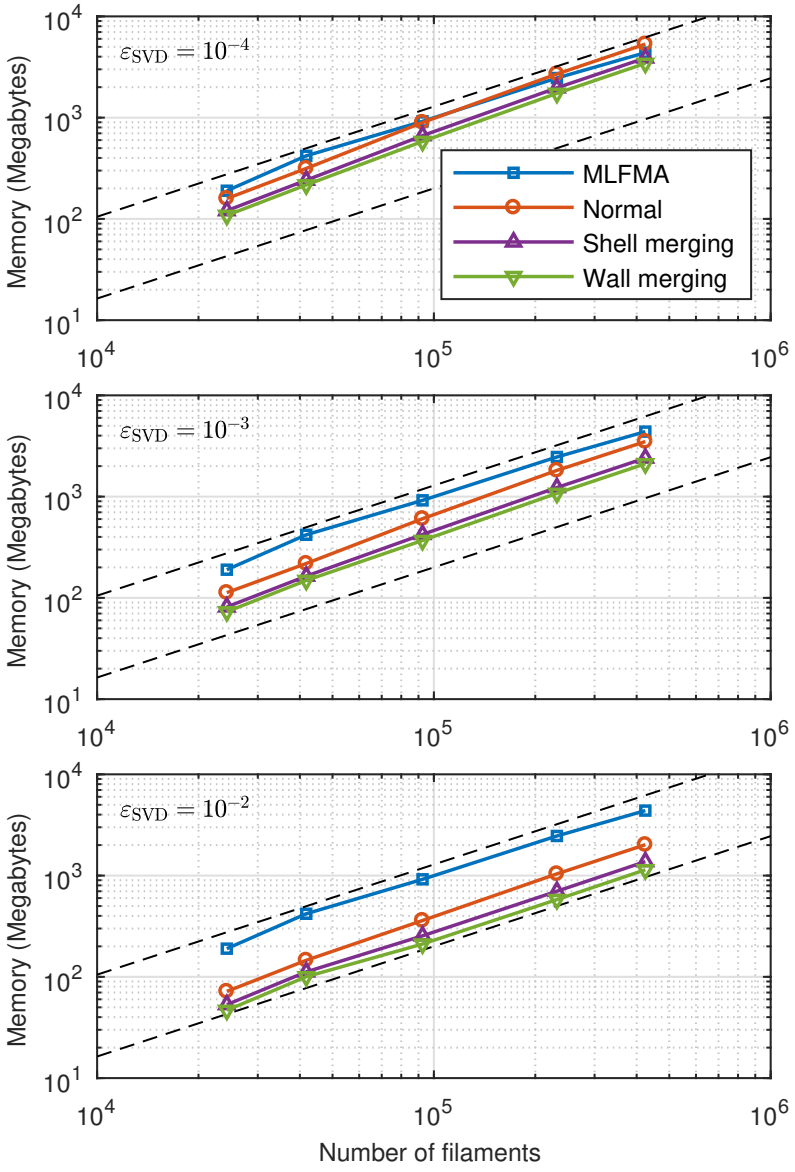

Figure 4.8: Memory required to store the mutual inductance matrix, versus number of filaments b. The dashed trend lines indicate $\mathcal{O}(b \log b)$ scaling. Top: results with $\varepsilon_{\mathrm{SVD}} = 10^{-4}$. Middle: results with $\varepsilon_{\mathrm{SVD}} = 10^{-3}$. Bottom: results with $\varepsilon_{\mathrm{SVD}} = 10^{-2}$.

Chapter 5

Conclusion

FastHenry is an inductance extraction programme that can be used for MQS analysis of superconducting structures, e.g. superconducting integrated circuits [4]. FastHenry uses an MLFMA solver for acceleration. In this work, the potential of the ACA algorithm as an efficient alternative to the MLFMA in FastHenry was investigated.

At first, ACA implementation was done on a single-level (Appendix A) [15]. The implementation was extended to a multilevel scheme with the MLACA (Appendix B) [16]. These results showed similar memory scaling for MLACA and MLFMA using the same interaction criteria. Further optimisation was achieved by introducing an MLACA-SVD solver (Appendix C) [9].

The MLACA-SVD solves the same underlying set of linear equations as the MLFMA, but has a different approach to calculating and storing the matrix entries. With the MLACA-SVD, near-interactions are calculated with rigorously determined semi-analytical procedures, where the accuracy can be controlled, rather than using FastHenry's existing, approximate calculations of fixed accuracy. The MLACA-SVD scheme compresses far interactions to evaluate the selected matrix entries needed for compression, by employing the new near-interaction procedures. For the compression itself, the SVD error tolerance can be set. This controls its accuracy at the cost of increased memory consumption. Two near-interaction criteria were tested for the MLACA-SVD, one being exactly the same as that of FastHenry's MLFMA (second-nearest neighbours) and the other being the nearest-neighbour criterion. The latter was found to yield more efficient results. The MLACA-SVD memory requirement for the compressed far interactions vs. number of filaments, was shown to scale practically identical to that of the MLFMA, for the complex and realistic superconducting integrated circuit models considered. The total memory requirement for the MLACA-SVD solver was shown to be lower than that of FastHenry's MLFMA, for the same relative error in the final output (port currents). The MLACA-SVD also provides full user control over approximation errors in the matrix representation, such that solutions can be obtained to any desired accuracy. The matrix setup time for the MLACA-SVD is longer than that of FastHenry's MLFMA, due to the former evaluating matrix entries more rigorously, as well as due to algorithmic differences [37]. However, the quadrature recipes could be further optimised and more generally, the MLACA-SVD matrix setup time can be readily reduced through parallelisation, to which the scheme is well suited.

After the MLACA-SVD was shown to be a viable alternative to the MLFMA in FastHenry, group merging was introduced as a method to further improve the efficiency (Appendix D) [10]. Motivation for the concept of merging was provided on the basis of the η-admissibility condition. Two group merging strategies were investigated, named wall merging and shell merging. Taking the analysis of a SQUID as an example, numerical results showed that both merging strategies yield results consistent with valid admissibility conditions. They reduce the required memory of the MLACA-SVD by a further approximately 30%. Shell merging was found to be marginally more efficient than wall merging, with regards to accuracy versus required memory. For results of similar accuracy as FastHenry's MLFMA, the MLACA-SVD solver with merging requires about 4 times less memory. For general volumetric structures (rather than the planar integrated circuits of interest here), merging could yield even larger memory reductions, due to more oppportunities for it. These merging strategies could be tested for other ACA solvers featuring the dynamic kernel, e.g. [19].

The efficient MLACA-SVD solver with group merging that has been developed during the course of this work, might in future be released on an open-source basis, such that FastHenry users worldwide may benefit from it.

Bibliography

[1] T. Orlando and K. A. Delin, *Foundation of Applied Superconductivity.* Addison-Wesley, 1991.

[2] K. K. Likharev and V. K. Semenov, "RSFQ logic/memory family: a new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," *IEEE Transactions on Applied Superconductivity*, vol. 1, no. 1, pp. 3–28, Mar 1991.

[3] T. S. Alstrøm, M. P. Sørensen, N. F. Pedersen, and S. Madsen, "Magnetic flux lines in complex geometry type-II superconductors studied by the time dependent Ginzburg-Landau equation," *Acta Applicandae Mathematicae*, vol. 115, no. 1, pp. 63–74, Jul 2011.

[4] M. Kamon, J. K. White, and M. J. Tsuk, "FastHenry: A multipole-accelerated 3-D inductance extraction program," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 9, pp. 1750–1758, 1994.

[5] A. E. Ruehli, "Equivalent circuit models for three-dimensional multiconductor systems," *IEEE Transactions on Microwave Theory and Techniques*, vol. 22, no. 3, pp. 216–221, Mar 1974.

[6] Y. Saad and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, no. 3, pp. 856–869, Jul. 1986.

[7] K. Nabors and J. White, "FastCap: A multipole accelerated 3-D capacitance extraction program," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 11, pp. 1447–1459, 1991.

[8] C. J. Fourie, "Full-gate verification of superconducting integrated circuit layouts with InductEx," *IEEE Transactions on Applied Superconductivity*, vol. 25, no. 1, pp. 1–9, Feb 2015.

[9] B. A. P. Nel and M. M. Botha, "An effcient MLACA-SVD solver for superconducting integrated circuit analysis," in preparation.

[10] ——, "MLACA with modified grouping strategy for efficient superconducting circuit analysis," *IEEE Transactions on Applied Superconductivity*, 2018, submitted for publication.

[11] J. R. Phillips and J. K. White, "A precorrected-FFT method for electrostatic analysis of complicated 3-D structures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 10, pp. 1059–1072, 1997.

[12] A. C. Yucel, I. P. Georgakis, A. G. Polimeridis, H. Bağcı, and J. K. White, "VoxHenry: FFT-accelerated inductance extraction for voxelized geometries," *IEEE Transactions on Microwave Theory and Techniques*, vol. 66, no. 4, pp. 1723–1735, 2018.

[13] N. Takeuchi, T. Ortlepp, Y. Yamanashi, and N. Yoshikawa, "Novel latch for adiabatic quantum-flux-parametron logic," *Journal of Applied Physics*, vol. 115, no. 10, 2014.

[14] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Mathematik*, vol. 86, no. 4, pp. 565–589, 2000.

[15] B. A. P. Nel and M. M. Botha, "Adaptive cross approximation (ACA) acceleration of superconducting circuit analysis," in *International Conference on Electromagnetics in Advanced Applications (ICEAA)*, 2017, pp. 1186–1189.

[16] ——, "Investigation of multilevel adaptive cross approximation (MLACA) acceleration for superconducting circuit analysis," in *International Conference on Electromagnetics in Advanced Applications (ICEAA)*, 2018, pp. 4–6.

[17] S. Kurz, O. Rain, and S. Rjasanow, "The adaptive cross-approximation technique for the 3-D boundary-element method," *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 421–424, 2002.

[18] M. Bebendorf and S. Rjasanow, "Adaptive low-rank approximation of collocation matrices," *Computing*, vol. 70, no. 1, pp. 1–24, 2003.

[19] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Transactions on Electromagnetic Compatibility*, vol. 47, no. 4, pp. 763–773, 2005.

[20] L. Grasedyck, "Adaptive recompression of $\mathcal{H}$-matrices for BEM," *Computing*, vol. 74, no. 3, pp. 205–223, 2005.

[21] M. Bebendorf and S. Kunis, "Recompression techniques for adaptive cross approximation," *Journal of Integral Equations and Applications*, vol. 21, no. 3, pp. 331–357, 2009.

[22] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. M.I.T. Press, Cambridge, Massachusetts, 1988.

[23] D. Cheng, *Field and Wave Electromagnetics*, ser. Addison-Wesley series in electrical engineering. Pearson Education Limited, 2013.

[24] J. Jin, *Theory and Computation of Electromagnetic Field*. John Wiley and Sons, 9 2010.

[25] Y. Saad, "Parallel iterative methods for sparse linear systems," in *Inherently Parallel Algorithms in Feasibility and Optimization and their Applications*, ser. Studies in Computational Mathematics, D. Butnariu, Y. Censor, and S. Reich, Eds. Elsevier, 2001, vol. 8, pp. 423 – 440.

[26] S. Rjasanow and O. Steinbach, *The fast solution of boundary integral equations*. Springer Science & Business Media, 2007.

[27] L. Grasedyck and W. Hackbusch, "Construction and arithmetics of $\mathcal{H}$-matrices," *Computing*, vol. 70, no. 4, pp. 295–334, 2003.

[28] M. Bebendorf and R. Grzhibovskis, "Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation," *Mathematical Methods in the Applied Sciences*, vol. 29, no. Jan, pp. 1721–1747, 2006.

[29] K. Frederix and M. Van Barel, "Solving a large dense linear system by adaptive cross approximation," *Journal of Computational and Applied Mathematics*, vol. 234, no. 11, pp. 3181–3195, 2010.

[30] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore: Johns Hopkins University Press, 1996.

[31] E. E. Tyrtyshnikov, "Tensor approximations of matrices generated by asymptotically smooth functions," *Sbornik: Mathematics*, vol. 194, no. 6, p. 941.

[32] P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration*, 2nd ed. Orlando, Florida: Academic Press, 1984.

[33] D. R. Wilton, S. M. Rao, A. W. Glisson, D. H. Schaubert, . M. Al-Bundak, and C. M. Butler, "Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains," *IEEE Transactions on Antennas and Propagation*, vol. 32, no. 3, pp. 276–281, 1984.

[34] I. V. Vernik, S. B. Kaplan, M. H. Volkmann, A. V. Dotsenko, C. J. Fourie, and O. A. Mukhanov, "Design and test of asynchronous eSFQ circuits," *Superconductor Science and Technology*, vol. 27, no. 044030, 2014.

[35] J. M. Tamayo, A. Heldring, and J. M. Rius, "Multilevel adaptive cross approximation (MLACA)," *IEEE Transactions on Antennas and Propagation*, vol. 59, no. 12, pp. 4600–4608, 2011.

[36] C. J. Fourie, C. Shawawreh, I. V. Vernik, and T. V. Filippov, "High-accuracy InductEx calibration sets for MIT-LL SFQ4ee and SFQ5ee processes," *IEEE Transactions on Applied Superconductivity*, vol. 27, no. 2, pp. 1–5, 2017.

[37] D. Brunner, M. Junge, P. Rapp, M. Bebendorf, and L. Gaul, "Comparison of the fast multipole method with hierarchical matrices for the Helmholtz-BEM," vol. 58, no. 2, pp. 131–158, 2010.

Appendix A

Conference paper – Single level ACA [15]

B. A. P. Nel and M. M. Botha, "Adaptive cross approximation (ACA) acceleration of superconducting circuit analysis," in 19th International Conference on Electromagnetics in Advanced Applications (ICEAA2017), Verona, Italy, September 2017, 4 pages.

Adaptive cross approximation (ACA) acceleration of superconducting circuit analysis

B. A. P. Nel* M. M. Botha†

***Abstract* — The open source package FastHenry is a magnetoquasistatic analysis program which can be used for inductance calculations of arbitrary three-dimensional superconductor circuit layouts. It employs the Partial Element Equivalent Circuit (PEEC) method, with multi-level, fast multipole method (MLFMM) acceleration of the linear solver. The adaptive cross approximation (ACA) algorithm is an alternative approach to the MLFMM, to accelerate integral equation based methods. The potential benefits of using the ACA as an efficient alternative to the MLFMM in FastHenry, is investigated. Single-level ACA results are presented. Results show that less directly-calculated near interactions need to be stored for the ACA.**

1 INTRODUCTION

The open source package FastHenry is a magnetoquasistatic analysis program used for inductance calculations of arbitrary three-dimensional conducting structures [1]. The applications often relate to integrated circuit layout. It employs the Partial Element Equivalent Circuit (PEEC) method, with multi-level, fast multipole method (MLFMM) [2] acceleration of the linear solver. The latter is made possible by the format of the underlying mutual inductance expression [1].

Superconductors are specific materials, e.g. niobium-titanium, that, when brought below a certain critical temperature, have no electrical resistance. Superconductivity is a quantum mechanical phenomenon. Superconducting Josephson Junctions switch fast and use considerably less power than semi-conductor transistors. In recent years, FastHenry has started to support the option to analyze superconducting circuits [3].

The adaptive cross approximation (ACA) [4, 5] is a well-established approach to accelerate the method of moments (MoM) for efficient full-wave electromagnetic analysis [6]. It is purely algebraic and therefore independent of the Greens function, but it does rely on the rank deficient property of the interaction between well-separated groups of basis functions, which follows from the nature of the Greens function. The matrix formulation used in FastHenry for superconducting circuit analysis which is currently accelerated with the MLFMM is also well-suited for the application of the ACA.

In this paper the potential benefits of using the ACA as an alternative to the MLFMM, as solver in the FastHenry code, is explored with regards to the storage requirements for the compressed, mutual inductance matrix. In Section 2 the FastHenry mutual inductance matrix and the ACA algorithm are briefly reviewed. Section 3 describes the test problems for which preliminary results are presented in Section 4.

2 ACA FOR MUTUAL INDUCTANCE MATRIX COMPRESSION

In FastHenry, the current distribution is discretized by dividing the volume occupied by the superconducting circuit paths into elongated right-angled hexahedral elements, also referred to as filaments [1]. The volume current density is assumed constant along each filament and directed along one of its principle axis, corresponding to its longest dimension. In the PEEC formulation used, non-local contributions are due to mutual inductance between filaments. The matrix entry relating to the mutual inductance between filaments i and j is calculated as follows:

$$L_{ij} = \frac{\mu}{4\pi a_i a_j} \int_{V_i} \int_{V_j} \frac{\mathbf{l}_i \cdot \mathbf{l}_j}{||\mathbf{r} - \mathbf{r}'||} dV' dV, \qquad (1$$

where V_i and V_j are the volume domains of filaments i and j, respectively. The vector $\mathbf{l}_i$ is the directional unit vector of current flow along filament i and similarly for $\mathbf{l}_j$; a_i and a_j are the cross sectional areas of filaments i and j, transverse to their respective directions of current flow.

The ACA is ideally suited to compress the mutual inductance matrix. After grouping filaments together based on spatial subdivision of the problem geometry, each sub-matrix relating to a non self inter-group interaction is considered in turn. The ACA iteratively factorizes each of these sub matrices. At every iteration k, a row vector V_i and a column vector U_k are added to the row matrix $\mathbf{V}_k$ and column matrix $\mathbf{U}_k$ respectively, based on eliminating estimated, maximum errors in the factorized representation of the k-th sub-matrix a

*Department of Electrical and Electronic Engineering, Stellenbosch University, Private Bag X1, Matieland, 7602 South Africa, e-mail: 17762944@sun.ac.za.

†Department of Electrical and Electronic Engineering, Stellenbosch University, Private Bag X1, Matieland, 7602 South Africa, e-mail: mmbotha@sun.ac.za.

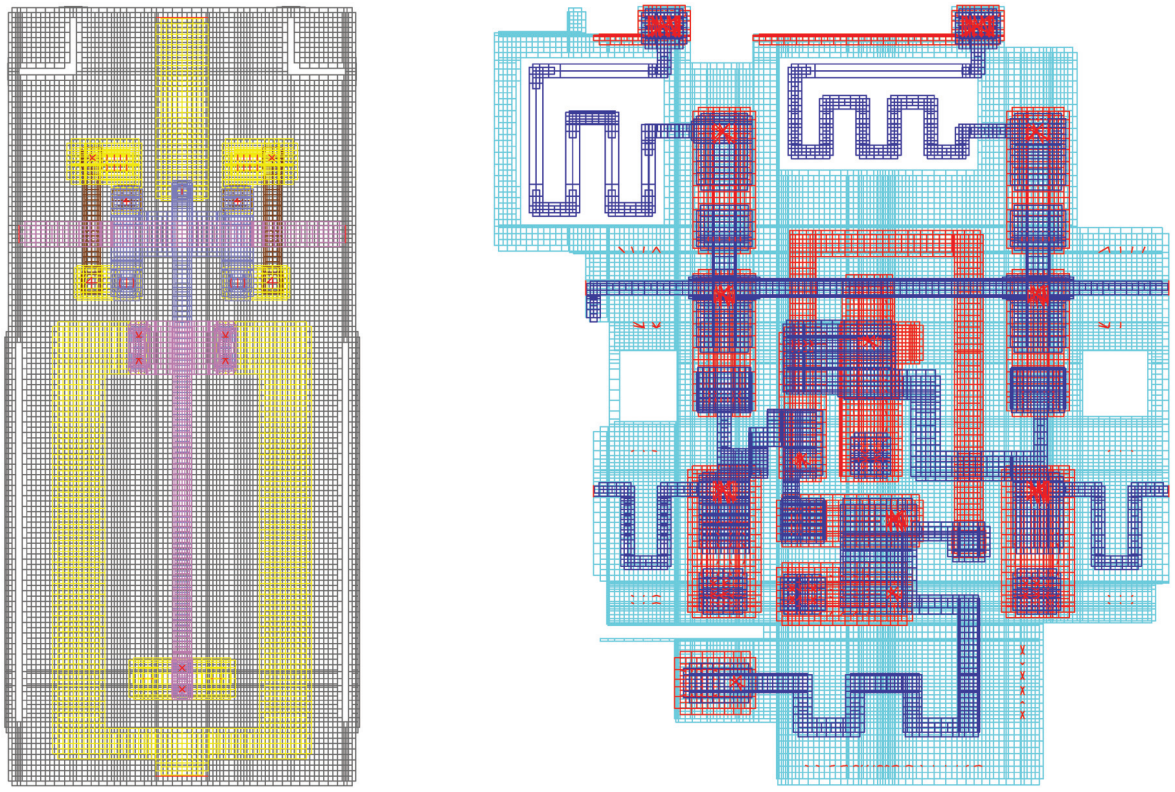

Figure 1: Test meshes. Left: small problem consisting of 23,226 filaments [7]. Right: large problem consistin of 43,183 filaments [8].

hand. With a specified error tolerance ϵ and the Frobenius norm, the termination criterion for the ACA algorithm is as follows:

$$||U_k|| \cdot ||V_k|| \leq \epsilon ||\mathbf{U}_k \mathbf{V}_k||. \tag{2}$$

3 TEST PROBLEMS

ACA compression for the mutual inductance matrices of two meshes representing integrated, superconducting circuits are investigated. Figures 1 shows the meshes; these will subsequently be referred to as the "small" problem and the "large" problem, respectively. The numbers of filaments are noted in the caption. The different colours represent different geometry layers. In the visualization of the large problem, a sky plane has been hidden for clarity. These meshes were produced with the InductEx package [9].

4 NUMERICAL RESULTS

All memory requirements reported in this section relate to double-precision representation of floating-point values.

As a benchmark, the memory required to store the mutual inductance matrices for the test problems, using the standard MLFMM solver of FastHenry, is noted in Table 1. The error tolerance level of the FastHenry MLFMM solver cannot be adjusted. Up to second-nearest neighbours are treated as near interactions.

Small	Large
133	309

Table 1: Memory in megabytes (MB), required fo the mutual inductance matrices, using the standar MLFMM solver of FastHenry. Six levels were automat ically employed by the solver, for both problems. Thes values include both near and far interaction storage.

For the ACA results, only a single-level imple mentation is considered. The error tolerance is var ied, the number of groups is varied and three dif ferent near-interaction criteria are considered (sel interactions only, up to nearest neighbours and u to second-nearest neighbours). Tables 2 and 3 lis the total memory required to store the mutual in ductance matrices for the various cases. Table shows which portions of the values in Tables 2 an 3 are due to near interaction storage.

The ACA is beneficial even for groups in clos proximity. Thus, clearly the most efficient near

Near-interaction criterion	Tolerance (ϵ)	32 groups	128 groups	528 groups
Self interactions	0.001	278	303	673
	0.01	215	180	450
	0.1	196	144	405
Nearest neighbours	0.001	1410	607	729
	0.01	1387	525	530
	0.1	1374	497	496
Second-nearest neighbours	0.001	2496	1096	843
	0.01	2485	1030	661
	0.1	2479	1013	640

Table 2: Small problem mutual inductance matrix memory required (MB), using ACA compression with variou algorithmic parameters.

Near-interaction criterion	Tolerance (ϵ)	49 groups	185 groups	689 groups
Self interactions	0.001	798	1001	2066
	0.01	589	499	1166
	0.1	528	399	1052
Nearest neighbours	0.001	3933	1818	2257
	0.01	3813	1384	1407
	0.1	3774	1312	1315
Second-nearest neighbours	0.001	7855	3124	2553
	0.01	7777	2748	1756
	0.1	7757	2699	1689

Table 3: Large problem mutual inductance matrix memory required (MB), using ACA compression with variou algorithmic parameters.

Near-interaction criterion	Small; #groups			Large; #groups		
	32	128	528	49	185	689
Self interactions	170	48	13	453	129	39
Nearest neighbours	1186	361	101	3264	932	282
Second-nearest neighbours	2298	887	258	7263	2340	676

Table 4: Memory required for storing the near interactions, corresponding to the results in Tables 2 and 3.

interaction criterion is to only store the self interactions directly. It can be seen that the benefit of factorizing larger inter-group interaction matrices, outweighs the additional storage requirement for self interactions in the case of $\epsilon = 0.001$, for the numbers of groups considered.

In [10] an ACA tolerance setting of $\epsilon = 0.001$ was found to yield good accuracy relative to required memory. It is seen that at this tolerance for these test problems, the best achieved memory requirements for the small and large problems are 278 MB and 798 MB, respectively. Thus, the single-level ACA manages storage for these problems of less than three times as much as the MLFMM (see Table 1), which is a multi-level scheme.

5 CONCLUSION

This paper investigates the potential of the ACA algorithm as an efficient alternative to the curren MLFMM solver within the FastHenry code, for su perconducting circuit analysis. Preliminary result demonstrate that the ACA could be a very efficien alternative. It is clear that less directly-calculate near interactions need to be stored in the case o the ACA. Only single-level ACA results are pre sented and the next step would be to extend th implementation to a multi-level scheme.

Acknowledgments

This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120.

References

[1] M. Kamon, J. K. White, and M. J. Tsuk, "FASTHENRY: A Multipole-Accelerated 3-D Inductance Extraction Program," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 9, pp. 1750–1758, 1994.

[2] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. M.I.T. Press, Cambridge, Massachusetts, 1988.

[3] "Whiteley Research Inc." [Online]. Available: http://www.wrcad.com.

[4] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Mathematik*, vol. 86, no. 4, pp. 565–589, 2000.

[5] S. Kurz, O. Rain, and S. Rjasanow, "The adaptive cross-approximation technique for the 3-D boundary-element method," *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 421–424, 2002.

[6] J.-M. Jin, *Theory and Computation of Electromagnetic Fields*. New York: John Wiley and Sons, 2010.

[7] N. Takeuchi, T. Ortlepp, Y. Yamanashi, and N. Yoshikawa, "Novel latch for adiabatic quantum-flux-parametron logic," *Journal of Applied Physics*, vol. 115, no. 10, p. 103910, 2014.

[8] I. V. Vernik, S. B. Kaplan, M. H. Volkmann, A. V. Dotsenko, C. J. Fourie, and O. A. Mukhanov, "Design and test of asynchronous eSFQ circuits," *Superconductor Science and Technology*, vol. 27, no. 4, p. 044030, 2014.

[9] C. J. Fourie, "Full-gate verification of superconducting integrated circuit layouts with InductEx," *IEEE Transactions on Applied Superconductivity*, vol. 25, no. 1, p. 1300209, 2015.

[10] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Transactions on Electromagnetic Compatibility*, vol. 47, no. 4, pp 763–773, 2005.

Appendix B

Conference paper – Multilevel ACA [16]

B. A. P. Nel and M. M. Botha, "Investigation of Multilevel Adaptive Cross Approximation (MLACA) Acceleration for Superconducting Circuit Analysis," in 20th International Conference on Electromagnetics in Advanced Applications (ICEAA2018), Cartagena de Indias, Colombia, September 2018, 4 pages.

Investigation of Multilevel Adaptive Cross Approximation (MLACA) Acceleration for Superconducting Circuit Analysis

1st Ben A. P. Nel
Department of Electrical and Electronic Engineering
Stellenbosch University
Stellenbosch, South Africa
17762944@sun.ac.za

2nd Matthys M. Botha
Department of Electrical and Electronic Engineering
Stellenbosch University
Stellenbosch, South Africa
mmbotha@sun.ac.za

***Abstract*—FastHenry is an open-source, three-dimensional, magneto-quasistatic solver applicable to superconducting, integrated circuit structures. The storage requirement for the dense, mutual inductance matrix grows quickly with problem size. This paper reports on progress with the development of a Multilevel Adaptive Cross Approximation (MLACA) solver as alternative to the existing Multilevel Fast Multipole Algorithm (MLFMA) in FastHenry, for compression of the mutual inductance matrix. Promising preliminary results show similar memory scaling for MLACA and MLFMA, when using the same near-interaction criterion. Further investigation and optimization of this MLACA solver for FastHenry is warranted.**

***Index Terms*—inductance extraction, integrated circuit (IC), partial element equivalent circuit (PEEC), superconductor**

I. Introduction

FastHenry [1] is an open-source, three-dimensional, magneto-quasistatic solver which is suitable for analysing conventional and superconducting circuit structures. The work presented in this paper builds upon previously reported work. In [2], preliminary results with a single-level Adaptive Cross Approximation (ACA) solver as alternative to the existing Multilevel Fast Multipole Algorithm (MLFMA) in FastHenry, were reported. In this paper, further progress is reported, with regards to extending the new solver to a Multilevel ACA (MLACA) scheme. The overall objective of this work is the efficient analysis of superconducting, integrated circuit structures.

The FastHenry solver is reviewed in Section II. Section III briefly explains how superconductivity is accounted for within the numerical formulation. The present status of the alternative, MLACA solver within FastHenry, is described in Section IV. Section V presents numerical results and Section VI concludes the paper.

The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

The National Research Foundation of South Africa also supported this work (Grant Number 96222).

II. The FastHenry Code

FastHenry [1] discretizes conducting regions into hexahedral filaments with a constant basis function associated with each filament, with current assumed to flow along the lengths (as defined for each filament). The magneto-quasistatic, volume integral equation formulation known as the Partial Element Equivalent Circuit (PEEC) method, is used. The discrete system $ZI_b = V_b$ relates the current coefficient vector I_b (b filaments in total) to the branch voltage vector V_b, through the branch impedance matrix $Z = R + j\omega L$. With i and j referring to specific filaments, the resistance and mutual inductance matrices are calculated as follows:

$$L_{ij} = \frac{\mu}{4\pi a_i a_j} \int_{V_i} \int_{V_j} \frac{\hat{\mathbf{l}}_i \cdot \hat{\mathbf{l}}_j}{||\mathbf{r} - \mathbf{r}'||} dV' dV, \tag{1}$$

$$R_{ii} = \frac{l_i}{\sigma(\mathbf{r}) a_i}, \tag{2}$$

with filament length l_i, orientation unit vector $\hat{\mathbf{l}}_i$, conductivity $\sigma(\mathbf{r})$ and cross sectional area a_i.

The linear system is reduced using a mesh current formulation:

$$MZM^T I_m = V_s, \tag{3}$$

with (typically sparse) source branch voltages vector V_s, mesh matrix $M \in \mathbb{R}^{m \times b}$ (with $MV_b = V_s$) and vector of mesh currents I_m, where m is the number of meshes [1]. The dense mutual inductance matrix L, requires storage of the order $\mathcal{O}(b^2)$. In order to reduce this storage requirement, the MLFMA is used [3]. An octree grouping scheme is employed. The resulting, compressed representation is used together with an iterative solution of the linear system.

III. Accounting for Superconductivity

Superconductivity results in the current having a superconducting component caused by the flow of Cooper-pairs [4].

Under superconducting conditions, the conductivity is defined as follows:

$$\sigma(\mathbf{r}) = \tilde{\sigma}_0(\mathbf{r}) + \frac{1}{j\omega\mu\lambda(\mathbf{r})^2}, \tag{4}$$

where $\tilde{\sigma}_0(\mathbf{r})$ is the temperature dependent conductivity and the second term on the right is as a result of the superconduction effect; λ is the temperature dependent London penetration depth. Substituting (4) into (2), accounts for superconductivity within the FastHenry code. The resistance matrix R, remains diagonal.

IV. MLACA Solver

The ACA algorithm is a purely algebraic approach which exploits rank-deficiency [5], [6]. It can be used to represent off-diagonal mutual inductance sub-matrices in terms of factors which are much cheaper to store. Blocks representing well-separated interactions have entries dependent on the inverse of the distance between elements. They are ideally suited for the application of the ACA. However, in (1) the orientation factor $\hat{\mathbf{l}}_i \cdot \hat{\mathbf{l}}_j$ can prevent the function from complying to the prerequisite of being asymptotically smooth, for ACA application [5]. The solution to this problem will be the subject of another publication by the present authors, but it is incorporated in the results presented here.

Applying ACA to a rank-deficient $m \times n$ matrix $A_{m\times n}$, results in a two-factor approximation $A_{m\times n} \approx U_{m\times k}V_{k\times n}$, where k is dependent on the rank of $A_{m\times n}$ and the specified relative error tolerance. Using a hierarchical octree subdivision of the mesh, interactions are factorized at the highest possible level, while regarding up to second-nearest neighbours as near-interactions which are not suitable for factorization on a given level. This near-interaction criterion is the same as used in the existing MLFMA solver of FastHenry. Benefits of applying MLACA as opposed to MLFMA, include the ease with which the error tolerance can be adjusted in order to obtain a desired accuracy, as well as the ability to exploit symmetry.

V. Results

Figure 1 shows the test geometry which represents a superconducting adiabatic quantum-flux-parametron logic (AQFP) gate [7]. This geometry is meshed uniformly with the number of filaments varied from $10,874$ to $714,315$; meshes are generated with InductEx [8]. Figure 2 shows how the memory requirement for the compressed part of the mutual inductance matrix, scales with the number of filaments; MLACA with varying relative factorization error tolerances, is compared with the MLFMA. All schemes scale similarly, with the MLFMA being the most efficient. In [6] a tolerance of 10^{-3} is deemed to be sufficiently accurate. With this tolerance, MLACA uses between 3.3 and 5.7 times the memory of MLFMA, when near-interaction storage is not considered. These results require further investigation with regards to the relative accuracies of the solutions, which are affected by various algorithmic aspects.

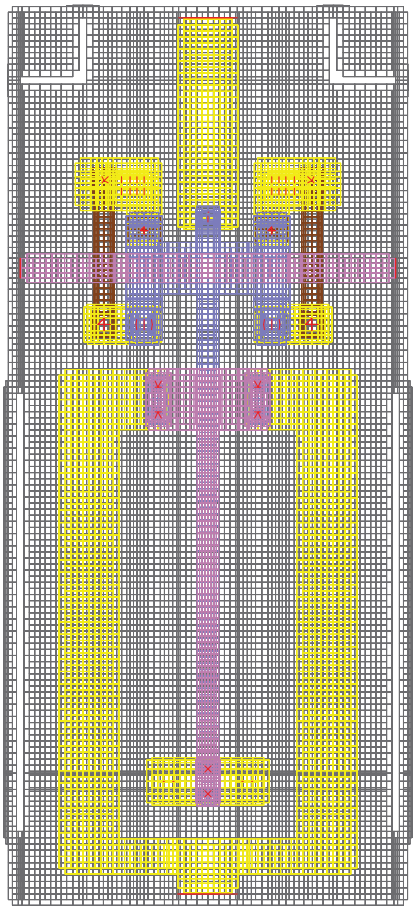

Fig. 1. Test geometry, which represents a superconducting AQFP gate [7].

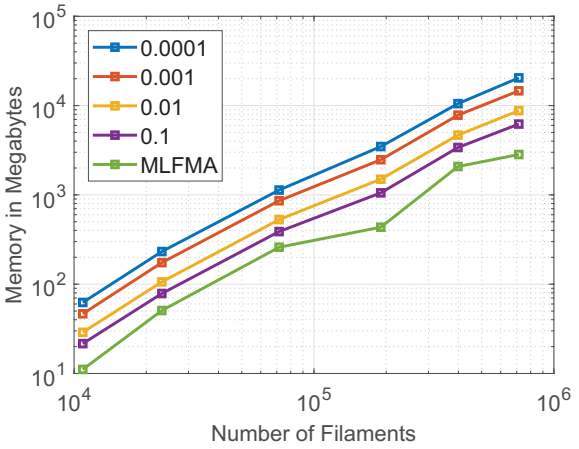

Fig. 2. Memory scaling for MLACA with varying relative factorization error tolerances, compared to MLFMA.

VI. Conclusion

Preliminary results show similar memory scaling for MLACA and MLFMA, using the same near-interaction criterion. These promising results serve as motivation for further optimization of this MLACA solver for FastHenry.

References

[1] M. Kamon, J. K. White, and M. J. Tsuk, "FASTHENRY: A Multipole-Accelerated 3-D Inductance Extraction Program," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 9, pp. 1750–1758, 1994.

[2] B. A. P. Nel and M. M. Botha, "Adaptive cross approximation (ACA) acceleration of superconducting circuit analysis," in *International Conference on Electromagnetics in Advanced Applications (ICEAA)*, 2017, pp. 1186–1189.

[3] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. M.I.T. Press, Cambridge, Massachusetts, 1988.

[4] T. Orlando and K. A. Delin, *Foundation of Applied Superconductivity*. Addison-Wesley, 1991.

[5] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Mathematik*, vol. 86, no. 4, pp. 565–589, 2000.

[6] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Transactions on Electromagnetic Compatibility*, vol. 47, no. 4, pp. 763–773, 2005.

[7] N. Takeuchi, T. Ortlepp, Y. Yamanashi, and N. Yoshikawa, "Novel latch for adiabatic quantum-flux-parametron logic," *Journal of Applied Physics*, vol. 115, no. 10, 2014.

[8] C. J. Fourie, "Full-gate verification of superconducting integrated circuit layouts with InductEx," *IEEE Transactions on Applied Superconductivity*, vol. 25, no. 1, 2015.

Appendix C

Journal paper – Efficient MLACA-SVD solver [9]

B. A. P. Nel and M. M. Botha, **"An efficient MLACA-SVD solver for superconducting integrated circuit analysis"**

An efficient MLACA-SVD solver for superconducting integrated circuit analysis

Ben A. P. Nel[1] **and Matthys M. Botha**[1]

1 Department of Electrical and Electronic Engineering, Stellenbosch University, South Africa

E-mail: 17762944@sun.ac.za, mmbotha@sun.ac.za

October 2018

Abstract. Inductance extraction for superconducting integrated circuits requires the accurate solution of structural current distributions. FastHenry is a well-known, magnetoquasistatic solver suitable for this task. It is based upon the partial element equivalent circuit (PEEC), integral equation method, with the structure discretized into hexahedral filaments. It employs the multilevel fast multipole algorithm (MLFMA) for compressed storage of the mutual inductance matrix, which accounts for most of the required memory. This MLFMA implementation is especially memory efficient, given certain approximations and algorithmic parameter choices. However, errors are introduced into the matrix representation. Here, a multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) is presented as alternative to FastHenry's existing MLFMA solver. MLACA-SVD compresses off-diagonal matrix blocks to a specified error tolerance, based on evaluating selected entries. Quadrature recipes are presented for guaranteed accuracy of matrix entry evaluation. Numerical results for examples of practical interest show that the MLACA-SVD memory scaling versus b (number of filaments) is practically identical to that of FastHenry's MLFMA, and is close to $\mathcal{O}(b \log b)$. The MLACA-SVD requires less memory for the same solution accuracy, and furthermore offers complete control over matrix approximation errors. For the examples considered, it is found to be a more efficient solver. It is well suited to parallelization.

Keywords: Acceleration, hexahedron, hierarchical octree, low frequency, low-rank factorization, near-interaction criterion, numerical integration, volume current density.

1. Introduction

To characterize the behaviour of complex three-dimensional superconducting integrated circuits, e.g. by constructing equivalent circuit models [1, 2], it is necessary to accurately solve the structural current distributions [3, 4, 5]. FastHenry is a well-known, efficient magnetoquasistatic solver which is suitable for inductance extraction [6]. It is based upon the partial element equivalent circuit (PEEC), integral equation method, with the structure discretized into right-angled hexahedral filaments. The mutual inductance matrix, representing inter-filament coupling, accounts for the bulk of the computer memory requirement. FastHenry employs the multilevel fast multipole algorithm (MLFMA) for compressed storage of this matrix. In this paper, a multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) is presented as alternative to the existing MLFMA solver, with the aim of improving the efficiency of FastHenry.

The scalar, static Green function features in FastHenry's mutual inductance matrix. The MLFMA is based upon analytical factorization of this function and hence, compression of the matrix. It is a very well established approach [7, 6]. Its implementation in FastHenry is especially memory efficient, given the approximations and fixed algorithmic parameter choices introduced. Theoretically, the compressed matrix storage scaling is expected to be $\mathcal{O}(b)$ instead of the conventional $\mathcal{O}(b^2)$ [6], where b denotes the number of filaments. The ACA is a purely algebraic approach to compression [8, 9, 10], which emerged after FastHenry was already established. ACA is applicable to asymptotically smooth kernels such as the static Green function of interest, for which the MLACA is expected to yield $\mathcal{O}(b \log b)$ memory scaling [8, 10, 11]. For an off-diagonal matrix block, the ACA determines an approximate rank and factorized form, by traversing rows and columns until convergence is reached, according to a specified error tolerance. The ACA algorithm has been extended with SVD recompression [12, 13]. The SVD is not simply applied directly (without ACA), although it would yield the optimal approximant, because it would require knowledge of the whole block and would require much higher runtime. Finally, it should be noted that there are also FFT-based acceleration (compression) schemes [14, 15], but those require a regular mesh, while the present work is focused on providing an alternative which is directly applicable to FastHenry meshes.

Section 2 reviews the existing FastHenry solver, with particular attention to aspects relevant to the work at hand. Section 3 explains how ACA-SVD is applied to compress a mutual inductance sub-matrix, representing the coupling between two disjoint groups of filaments. Given that the control of compression accuracy is an important feature of the ACA-SVD, it was found necessary to revisit the accuracy to which the direct calculation of matrix entries is performed, such that the benefits of accurate compression is not lost due to large quadrature errors. This lead to the development of quadrature recipes for guaranteed matrix entry accuracy, presented in Section 4. Section 5 describes the complete MLACA-SVD solver scheme, which is followed by numerical results in Section

6, to assess the performance of the new solver, for superconducting integrated circuit modelling. Section 7 delivers the overall conclusions to this work.

2. The MLFMA-accelerated, integral equation based, PEEC formulation of FastHenry

In FastHenry, the conducting structure of interest is discretized into a mesh with a total of b, right-angled hexahedral filaments, with a single, constant axial basis function upon each one. Let I_b denote the vector of complex-valued, phasor current coefficients, representing the currents flowing along all filaments, with reference directions according to the basis functions. The partial element equivalent circuit (PEEC) method, based on a volume integral equation with assumed magnetoquasistatic conditions, then yields

$$ZI_b = V_b, \tag{1}$$

where V_b is the vector of filament potential differences, with $Z = R + j\omega L$ being the $b \times b$ branch impedance matrix; ω is the angular frequency. Galerkin testing is used. The resistance and mutual inductance matrices are defined as

$$R_{ii} = \frac{\ell_i}{\sigma(\boldsymbol{r})a_i} \tag{2}$$

$$\sigma(\boldsymbol{r}) = \sigma_{\mathrm{C}}(\boldsymbol{r}) + \frac{1}{j\omega\mu\lambda(\boldsymbol{r})^2} \tag{3}$$

$$L_{ij} = \frac{\mu}{4\pi a_i a_j} \int_{V_i} \int_{V_j} \frac{\hat{\ell}_i \cdot \hat{\ell}_j}{|\boldsymbol{r} - \boldsymbol{r}'|} dV' dV \tag{4}$$

with $i, j \in \{1, ..., b\}$; i and j refer to the testing and source filament numbers, respectively. Equation (3) defines the conductivity, with σ_{C} denoting the temperature-dependent, conventional conductivity; the second term accounts for superconductivity [16], with λ being the temperature-dependent London penetration depth. In (2) and (4), V_i, a_i, ℓ_i and $\hat{\ell}_i$ denote the i-th filament's domain, cross-sectional area, axial length and axial unit-vector, respectively (similarly for j). The basis function associated with the i-th filament is $\hat{\ell}_i/a_i$.

A mesh-current approach yields the final system of linear equations:

$$MZM^T I_m = V_s, \tag{5}$$

where m is the total number of linearly independent loops in the mesh and V_s is the (typically sparse) vector of source voltages in all loops, such that $MV_b = V_s$, with M following from Kirchhoff's voltage law applied to all loops. The loop currents relate to the filament currents as $M^T I_m = I_b$.

The resistance matrix is highly sparse (diagonal), but the mutual inductance matrix is dense and storage requirements are prohibitive for large b, which is typically the case for superconducting, integrated circuit structures. Therefore, the well-known Multilevel Fast Multipole Algorithm (MLFMA) is used to represent L in a factorized format, theoretically requiring $\mathcal{O}(b)$ storage and $\mathcal{O}(b)$ runtime for a matrix-vector product

[6]. An iterative solver using the generalized minimal residual (GMRES) method [17] combined with preconditioning is used to solve (5) [6].

To employ the MLFMA, start by defining the diagonal, vector-valued *basis matrix*, with diagonal entries equal to the basis functions evaluated within each element, multiplied with the elemental volume:

$$\boldsymbol{\Lambda}_{ii} = \frac{V_i \hat{\ell}_i}{a_i} = \ell_i \hat{\ell}_i \qquad i \in \{1, ..., b\}. \tag{6}$$

Furthermore, define the *potential matrix* Φ:

$$\Phi_{ij} = \frac{\mu}{4\pi V_i V_j} \int_{V_i} \int_{V_j} \frac{1}{|\boldsymbol{r} - \boldsymbol{r}'|} dV' dV \qquad i, j \in \{1, ..., b\}. \tag{7}$$

It follows that the mutual inductance matrix can be expressed as

$$L = \boldsymbol{\Lambda} \cdot \Phi \boldsymbol{\Lambda}. \tag{8}$$

An MLFMA-representation is established for Φ. The MLFMA-representation rests upon a hierarchical (multilevel), octree grouping of the mesh filaments. Non-self interactions between groups relating to off-diagonal blocks in Φ, are represented in factorized form by exploiting a truncated, series expansion representation of the Green function $1/|\boldsymbol{r}-\boldsymbol{r}'|$ [7]. Factorization involves aggregation, translation and disaggregation factors. Aggregation and disaggregation factors can be utilized at multiple levels and the objective is to treat interactions at the highest possible level, at which they qualify as far interactions, according to a near-interaction criterion. This criterion is necessary to ensure accuracy of the truncated series expansion. A cubic, second-nearest neighbour criterion is used, as illustrated in Figure 6 (right). This means that interactions between groups at leaf level (the smallest group size), are stored directly in case both fall within a $3 \times 3 \times 3$ leaf-level group sized cube.

The accuracy with which this approach represents the true mutual inductance matrix will be evaluated later-on. The near-interaction entries are directly calculated using approximate analytical expressions or quadrature, as automatically determined to be most appropriate. Errors are introduced by the MLFMA into the far-interaction entries, due to the factorized Green function representation itself, as well as due to the factorization being done for a midpoint-quadrature based, approximate representation of Φ_{ij}, i.e.

$$\Phi_{ij} \approx \frac{\mu}{4\pi} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} \tag{9}$$

where $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ denote the centroids of volumes V_i and V_j, respectively. The reason for this is that the MLFMA implementation is most efficient when dealing with interactions between point sources (i.e. the midpoints), rather than interactions between distributed sources.

3. ACA-SVD compression of a mutual inductance, off-diagonal sub-matrix

Consider a sub-matrix denoted by L^{sub}, of the mutual inductance matrix (4), representing the interaction between two disjoint groups of filaments, denoted as groups P (consisting of p testing basis functions) and Q (consisting of q source basis functions). It follows that

$$L^{\text{sub}} = \boldsymbol{\Lambda}_P \cdot \Phi_{PQ} \boldsymbol{\Lambda}_Q, \tag{10}$$

where Φ_{PQ} represents the appropriate sub-matrix of the global potential matrix (7); and $\boldsymbol{\Lambda}_P$ and $\boldsymbol{\Lambda}_Q$ are appropriate, diagonal sub-matrices of the global basis matrix (6).

Generally, the ACA is expected to perform well for inter-group matrices of which the entries are proportional to Green functions associated with various physical boundary value problems [18]; particularly so for the asymptotically smooth, scalar Green function of Poisson's equation, as featured in (4). Performance improves as the distance between groups grow, relative to the group diameters [8, 19, 12]. Scalar testing/source functions present no difficulties, as they effectively just scale matrix rows and columns, which can be normalized. However, irregular scaling of matrix entries within rows and columns, can cause catastrophic algorithm breakdown [20, 21]. Such irregular scaling is encountered in L^{sub}, due to the dot-product between the axial unit-vectors. E.g. suppose in the two groups each filament is oriented in one of two orthogonal directions, then with appropriate numbering of degrees of freedom, L^{sub} takes the form

$$L^{\text{sub}} = \begin{bmatrix} L_{11}^{\text{sub}} & 0 \\ 0 & L_{22}^{\text{sub}} \end{bmatrix}. \tag{11}$$

If the ACA algorithm starts by investigating a column intersecting with L_{11}^{sub}, then the algorithm will continue to place crosses centred inside L_{11}^{sub}, until convergence. The part L_{22}^{sub} will never be examined and will be falsely considered to be zero. Therefore, as in the case of the MLFMA compression, the ACA is not applied directly to L^{sub} in (10), but rather to Φ_{PQ}.

Application of the standard ACA algorithm [8, 11], yields the following approximate factorization:

$$[\Phi_{PQ}]_{p\times q} \approx [U]_{p\times k} \left[V^T\right]_{k\times q}. \tag{12}$$

The algorithm requires evaluating selected rows and columns of Φ_{PQ}. In contrast to FastHenry's MLFMA, midpoint-integration is not used, but rather a quadrature recipe from Section 4, which ensures the rigorous evaluation of all matrix entries to a higher accuracy than a specified error tolerance.

The ACA-estimated rank $k \leq \min\{p, q\}$ (with the expectation that $k \ll \min\{p, q\}$ should hold) is determined by the factorization error tolerance setting. During the ACA factorization algorithm, the relative factorization error ε_k is approximately determined at each iteration k, by way of efficient recursive calculations [9]. It is defined and approximated as

$$\varepsilon_k \equiv \frac{\|\Phi_{PQ} - U_k V_k^T\|_F}{\|\Phi_{PQ}\|_F}$$

$$\approx \frac{\|U_k V_k^T - U_{k-1} V_{k-1}^T\|_F}{\|U_k V_k^T\|_F} = \frac{\|U_k(:,k)\|_F \|V_k(:,k)\|_F}{\|U_k V_k^T\|_F}, \tag{13}$$

where U_k and V_k denote the factors after k iterations; $\|\cdot\|_F$ represents the Frobenius norm. The algorithm terminates when a specified relative error tolerance level ε_{ACA}, is reached, i.e.

$$\varepsilon_k \leq \varepsilon_{\text{ACA}}. \tag{14}$$

Generally, the columns of neither U nor V are orthogonal [13]. Therefore, SVD is further applied to eliminate any possible redundancy within U and V, as recommended in [12, 13]. First, QR decompositions of U and V are computed using the Householder algorithm [22], as

$$U = [Q_U]_{p\times k}\,[R_U]_{k\times k} \tag{15}$$

$$V = [Q_V]_{q\times k}\,[R_V]_{k\times k}\,. \tag{16}$$

Then apply SVD to $R_U R_V^T$:

$$R_U R_V^T = \tilde{U}\Sigma\tilde{V}^T. \tag{17}$$

This yields the desired SVD of the product UV^T:

$$UV^T = Q_U\tilde{U}\Sigma\left(Q_V\tilde{V}\right)^T. \tag{18}$$

The singular values are found in descending order on the diagonal of Σ, with the largest denoted by $\sigma_{\max}$. All values falling below a relative threshold are removed [12]. The threshold is defined as

$$\frac{\sigma_i}{\sigma_{\max}} \leq \varepsilon_{\text{SVD}} \qquad i \in \{1, ..., k\}. \tag{19}$$

This results in a new rank $\tilde{k} \leq k$ and recompression of the original U and V matrices. Since the SVD rank is precise, while the ACA rank is approximate, a $10\times$ buffer factor is employed to ensure reliable ACA-SVD accuracy:

$$\varepsilon_{\text{SVD}} = 10\varepsilon_{\text{ACA}}. \tag{20}$$

The SVD tolerance is then considered the actual matrix compression tolerance.

The additional computational cost of adding SVD on top of ACA does not change the overall cost scaling of the ACA. The runtime for steps (15), (16) and (18) together, scales as $\mathcal{O}(k^2(p+q+k))$ [22].

4. Accurate evaluation of mutual inductance matrix entries

A major benefit of ACA-SVD compression is control over accuracy, by way of (19). For this to be of value, the errors in evaluating the mutual inductance matrix entries must also be controlled. Quadrature recipes have thus been developed which guarantee specified bounds on the relative errors with which the matrix entries are evaluated. Relative error is defined as

$$\varepsilon_{\text{quad}} = \left|\frac{L_{ij}^{\text{quadrature}} - L_{ij}^{\text{reference}}}{L_{ij}^{\text{reference}}}\right|. \tag{21}$$

Throughout this section, the testing and source basis functions are assumed to be co-linear, without loss of generality.

For the inner (source) integral in (4), either Gaussian (Gaussian-Legendre) quadrature [23] in product-rule format is used, or analytical evaluation [24]. The choice depends upon the distance to the testing domain, since Gaussian quadrature requires a smooth integrand. As the testing and source domains get closer, the near-singular (less smooth) behaviour of the kernel increases.

For the outer (testing) integral, Gaussian product-rule quadrature is always used. This is an optimal choice with regards to error convergence, which follows from considering the most extreme behaviour of the integrand, which occurs when V_i and V_j are equal (self-interaction) or share a face. It can be shown that the integrand's leading term behaves as $\mathcal{O}(\delta)$ when $\delta \to 0$, with δ denoting the orthogonal distance of the outer integration point from the coincident face. Since Gaussian quadrature is constructed for integrating polynomials, it is thus well-suited to this integrand.

The following normalized distance will be used in defining the recipes:

$$D_{\text{norm}} = \frac{D_{\text{min}}}{d_{\text{max}}} \tag{22}$$

with D_{min} denoting the minimum distance between the bounding boxes (in the global coordinate system) of V_i and V_j, and d_{max} denoting the maximum edge length among all six edge lengths in V_i and V_j. Tables 1, 2 and 3 show the recipes for achieving relative error levels of $\varepsilon_{\text{quad}} < 10^{-6}$, $\varepsilon_{\text{quad}} < 10^{-4}$ and $\varepsilon_{\text{quad}} < 10^{-2}$, respectively. These error levels are guaranteed up to elemental maximum aspect ratios of 1:100. To demonstrate the use of the tables, consider two elemental domains with respective aspect ratios 1:5:10 and 1:1:20, and with $D_{\text{norm}} = 1.5$. Then, to achieve $\varepsilon_{\text{quad}} < 10^{-6}$ it follows from Table 1 that Gaussian product-rule quadrature should be used on both domains, with respective orders $4 \times 6 \times 6$ and $4 \times 4 \times 6$. Figures 1, 3, 4 and 5 show quadrature error results using the proposed recipes, for two identical elemental domains under various relative positioning conditions. The elemental dimensions are chosen to be rather extreme, namely $1 \times 10 \times 100$. These results are invariant with regards to uniform scaling, thus no units are indicated. The results show that the recipes do indeed yield relative errors below the indicated thresholds.

Finally, it should be noted that more efficient recipes yielding less over-accuracy (as displayed in the example results), in exchange for fewer quadrature points, are possible. This could be done by introducing more D_{norm}-range brackets, more edge-length range brackets, and by determining the rule on each element based on D_{min} relative to its own maximum edge length, rather than d_{max}. Such further refinements are beyond the scope of this paper and can be introduced as part of code-optimization, as desired.

Table 1. Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-6}$. In case of Gaussian inner and outer quadrature, the orders are determined for the inner and outer elements individually and in the same way. I.e. for a given element, find its minimum edge length (denoted by ℓ_{min}) and assign an order to each dimension of that element, according to edge length ℓ, along that dimension.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension.		
			$\ell = \ell_{\text{min}}$	$\ell_{\text{min}} < \ell \leq 10\ell_{\text{min}}$	$10\ell_{\text{min}} < \ell \leq 100\ell_{\text{min}}$
Self	Gaussian	Analytic	7	12	27
< 0.1	Gaussian	Analytic	7	13	31
< 0.5	Gaussian	Analytic	5	8	10
< 2.5	Gaussian	Gaussian	4	6	6
< 12	Gaussian	Gaussian	3	4	4
< 450	Gaussian	Gaussian	2	2	2
≥ 450	Gaussian	Gaussian	1	1	1

Table 2. Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-4}$. In case of Gaussian inner and outer quadrature, the orders are determined for the inner and outer elements individually and in the same way. I.e. for a given element, find its minimum edge length (denoted by ℓ_{min}) and assign an order to each dimension of that element, according to edge length ℓ, along that dimension.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension.		
			$\ell = \ell_{\text{min}}$	$\ell_{\text{min}} < \ell \leq 10\ell_{\text{min}}$	$10\ell_{\text{min}} < \ell \leq 100\ell_{\text{min}}$
Self	Gaussian	Analytic	4	7	14
< 0.15	Gaussian	Analytic	4	8	18
< 0.5	Gaussian	Analytic	3	5	5
< 3	Gaussian	Gaussian	3	4	4
< 4	Gaussian	Gaussian	2	2	2
< 41	Gaussian	Gaussian	1	2	2
≥ 41	Gaussian	Gaussian	1	1	1

Table 3. Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-2}$. In case of Gaussian inner and outer quadrature, the orders are determined for the inner and outer elements individually and in the same way. I.e. for a given element, find its minimum edge length (denoted by ℓ_{min}) and assign an order to each dimension of that element, according to edge length ℓ, along that dimension.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension.		
			$\ell = \ell_{\text{min}}$	$\ell_{\text{min}} < \ell \leq 10\ell_{\text{min}}$	$10\ell_{\text{min}} < \ell \leq 100\ell_{\text{min}}$
Self	Gaussian	Analytic	2	3	3
< 0.25	Gaussian	Analytic	1	3	6
< 3.5	Gaussian	Gaussian	1	3	4
≥ 3.5	Gaussian	Gaussian	1	1	1

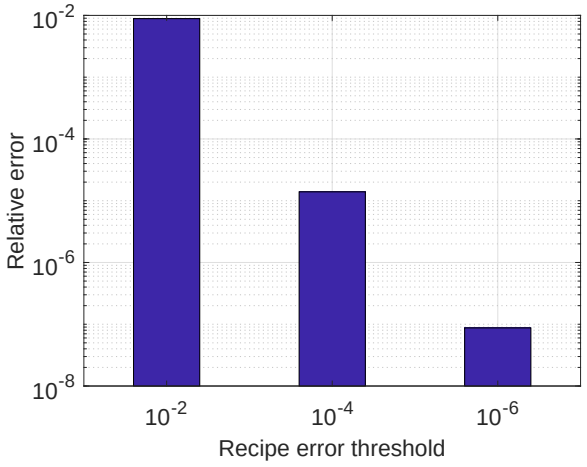

Figure 1. Self-interaction (L_{ii}) quadrature errors using the three recipes, for a $1 \times 10 \times 100$ element.

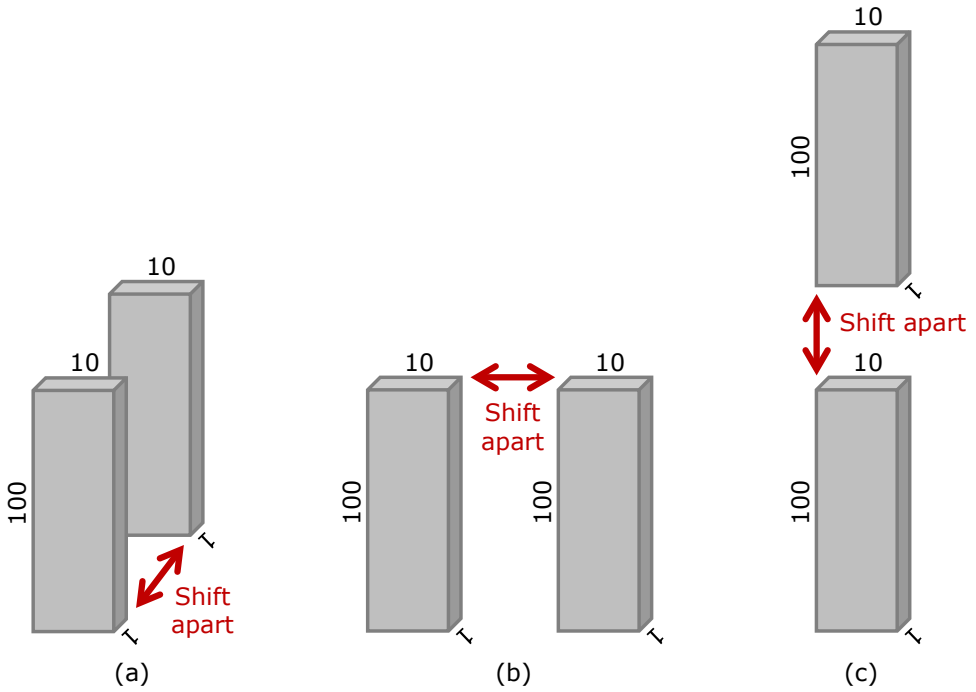

Figure 2. Figurative illustration of the experimental set-ups for the results in Figures 3, 4 and 5. (a) Shifting apart in direction of the elemental 1-dimensions. (b) Shifting apart in direction of the elemental 10-dimensions. (c) Shifting apart in direction of the elemental 100-dimensions.

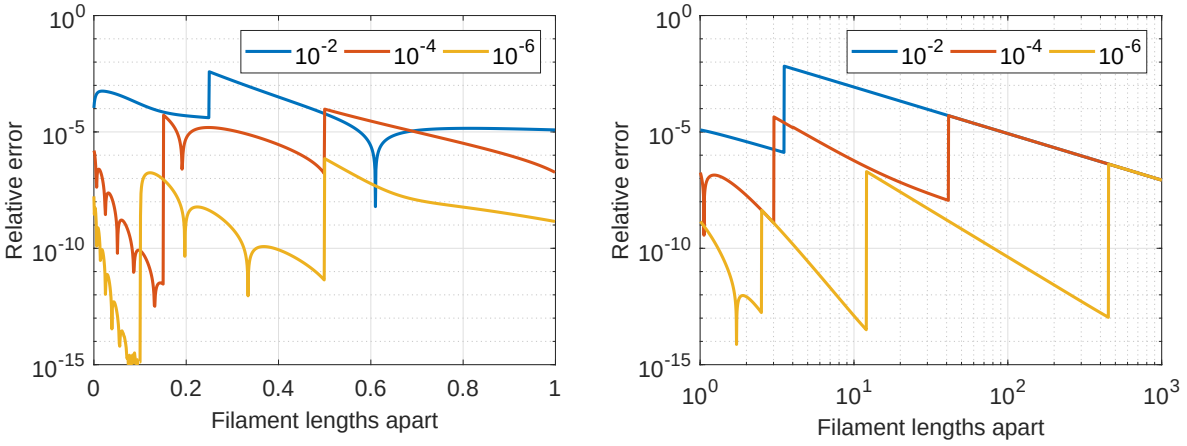

Figure 3. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 10×100 faces, with separation introduced orthogonally to these faces, in the 1-dimensions direction (see Figure 2(a)).

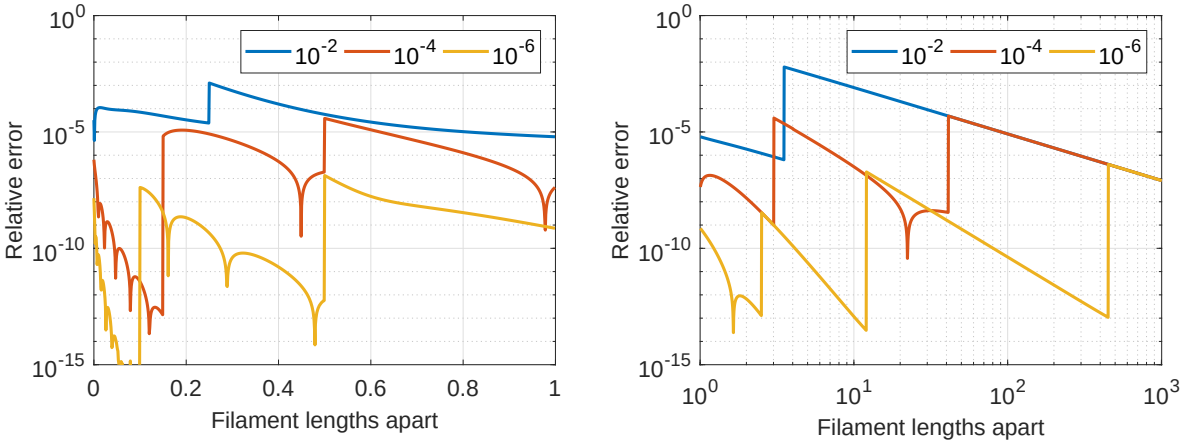

Figure 4. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 1×100 faces, with separation introduced orthogonally to these faces, in the 10-dimensions direction (see Figure 2(b)).

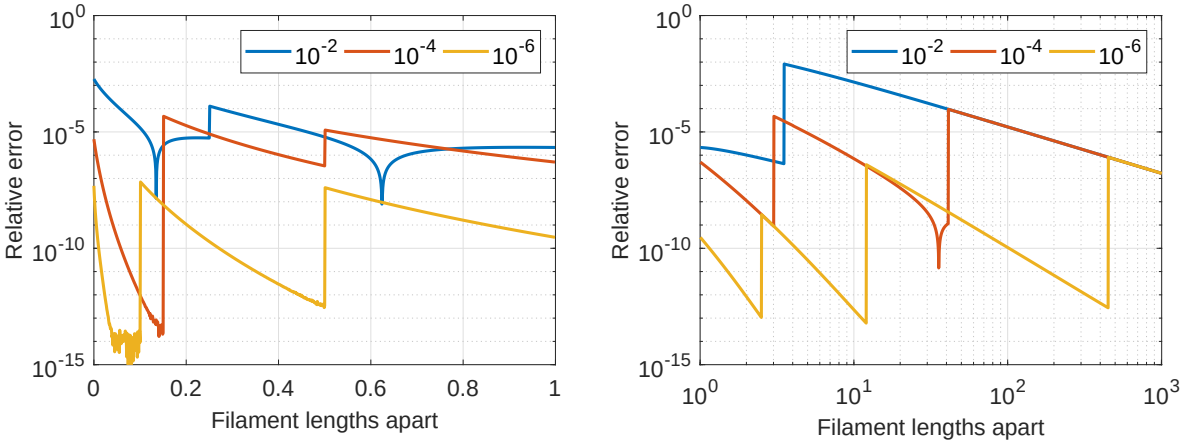

Figure 5. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 1×10 faces, with separation introduced orthogonally to these faces, in the 100-dimensions direction (see Figure 2(c)).

5. The MLACA-SVD solver

In Section 3, the compression of an off-diagonal block of Φ is presented, which is the core function employed in the MLACA-SVD solver. The solver employs the existing hierarchical (multilevel), octree mesh grouping scheme from FastHenry's MLFMA. Similarly to the MLFMA, inter-group interactions are classified as far or near interactions. Also similarly, the MLACA-SVD solver compresses all far interactions, with the objective to treat interactions at the highest possible level. What is different from FastHenry's MLFMA, apart from the compression algorithm, is the near-interaction criterion, the symmetric storage scheme and the calculation of near-interaction matrix entries.

For the ACA on a given level of the octree, the η-admissibility condition, which predicts exponential decay of singular values for asymptotically smooth kernels [8, 19], can be stated as follows:

$$\operatorname{diam}(D_Q) \leq \eta \operatorname{dist}(D_P, D_Q) \qquad \{0 < \eta\}, \tag{23}$$

where D_P and D_Q are taken as the octree cubes (identical in size) defining the testing and source filament groups, respectively. Increasing the distance between D_P and D_Q results in more rapid singular value decay, therefore reducing η is regarded as strengthening the admissibility condition. Figure 6 illustrates two near-interaction criteria, depicted in two dimensions for simplicity. FastHenry's MLFMA uses the second-nearest neighbour criterion exclusively. The MLACA-SVD is tested with this same criterion. It corresponds to $\eta < \sqrt{3}/2$, which is a strong admissibility condition. The MLACA-SVD is also tested with the nearest-neighbour criterion. It is a weaker admissibility condition, with $\eta < \sqrt{3}$, which should theoretically result in a less accurate ACA approximation for the same rank [12]. The benefit of the nearest-neighbour criterion is that valid far interactions generally occur at a higher level, allowing ACA-SVD compression of larger sub-matrices. In Section 6 the relative efficiencies of these criteria are evaluated.

Since the MLACA-SVD compresses individual matrix blocks independently, the symmetric nature of L can be exploited. Only the upper-triangular part is factorized and stored. This information is then used both directly and in transposed form, when calculating a matrix-vector product. Symmetry is also exploited in the storage of near-interaction matrix entries.

To determine the near-interaction matrix entries resulting from leaf-level near-interactions, the MLACA-SVD solver does not use the direct-calculation routines of FastHenry. Rather, the quadrature recipes presented in Section 4 are used; and with the same quadrature error threshold as that for constructing the ACA-SVD factorizations, such that accuracy is fully controlled.

Finally, with regards to the matrix equation solver: exactly the same preconditioner and GMRES iteration scheme as for the MLFMA solver (noted in Section 2), are used for the MLACA-SVD.

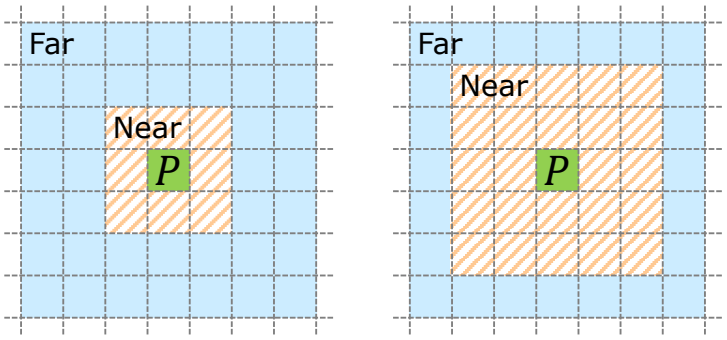

Figure 6. Two near-interaction criteria, with respect to group P. Left: nearest neighbours. Right: second-nearest neighbours.

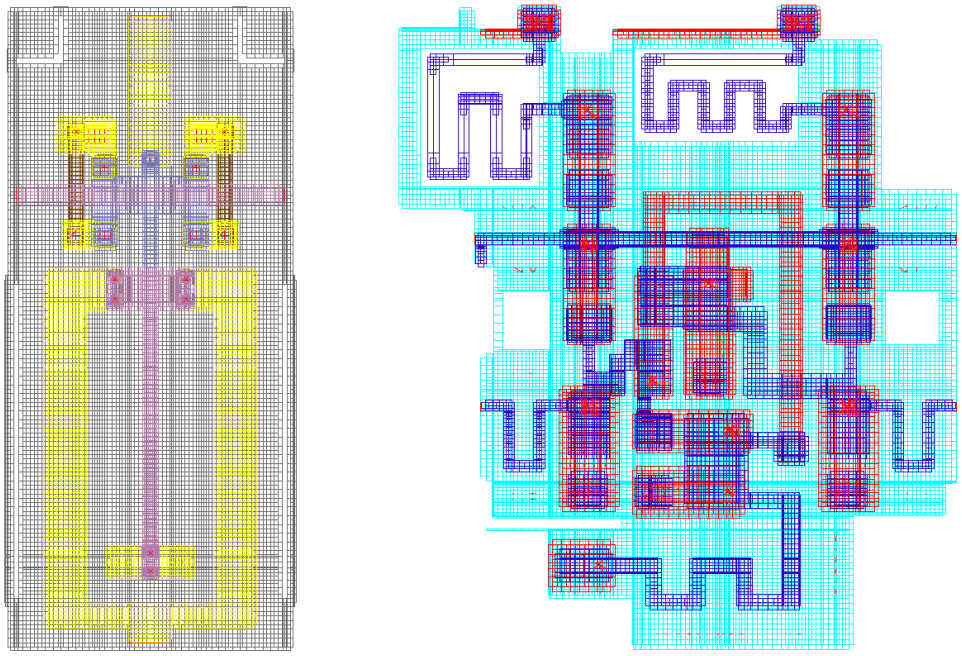

Figure 7. Example meshes of the two test models, representing superconducting, integrated circuit structures. Left: an AQFP gate. Right: an eSFQ circuit.

6. Results

Figure 7 shows the two superconducting, integrated circuit test models used to evaluate the performance of the MLACA-SVD solver; these are an adiabatic quantum-flux-parametron (AQFP) gate [25] and an energy-efficient single flux quantum (eSFQ) circuit [26]. Performance with regards to memory requirement and accuracy with which the true matrix L is approximated by the compressed versions, is considered.

All MLACA-SVD results are obtained using the $\varepsilon_{\text{quad}} < 10^{-6}$ quadrature recipe from Section 4, both for obtaining the factorization and the near-interaction matrix entries. The label "MLFMA" refers to results obtained with FastHenry's MLFMA solver with default settings, which employs the standard FastHenry routines to evaluate the near-interaction matrix entries.

6.1. Memory

The total memory required to store the mutual inductance matrix L (4) in compressed form, is measured as the mesh density is varied. Meshing is done with the InductEx package [2]. For the MLACA-SVD solver the near-interaction criteria of Figure 6 are both considered, as well as four factorization tolerance settings $\varepsilon_{\text{SVD}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Figures 8 and 9 show the memory scaling results for the two models.

Firstly, consider the observed scaling orders. Dashed trend lines of $\mathcal{O}(b \log b)$ show that the MLACA-SVD versions scale at this rate or slightly above it. This is expected for static/low-frequency solutions with asymptomatically smooth kernels [11]. According to [6], the expected scaling of the MLFMA is $\mathcal{O}(b)$ and here it does seem to scale on average, a little better than the MLACA-SVD versions. However, note that the MLFMA does not exploit symmetry in the storage of its directly-calculated, near-interaction entries. At low mesh sizes these tend to dominate the MLFMA memory requirement. To remove this influence, the MLFMA is compared with the second-nearest neighbour criterion MLACA-SVD, with near-interaction storage excluded for both, i.e. purely the compressed storage of exactly the same sets of matrix entries are compared. Figure 10 shows the results, which indicate that for the considered test cases, the MLFMA in fact scales the same or even slightly worse than the MLACA-SVD.

Secondly, consider the comparative memory requirements of Figures 8 and 9 in absolute terms. Clearly, applying the nearest-neighbour criterion MLACA-SVD always yields a lower memory requirement than with the second-nearest criterion version, for the same mesh and SVD tolerance. Also clearly, the MLACA-SVD memory requirement is strongly and uniformly influenced by the choice of SVD tolerance. Arguably the most important observation, is that the nearest-neighbour criterion MLACA-SVD outperforms the MLFMA. However, this should be considered in conjunction with the relative accuracies of these compressed representations. Accuracy is considered next.

6.2. Accuracy

Both test models have a number of defined ports. At each port in turn, a unit voltage source is connected with all others shorted out. After solving the global current distribution, a vector of the ports currents is obtained. These vectors form the columns of a port current matrix I_{port}, which is the key input to circuit parameter extraction tools such as InductEx [2].

Accuracy of the compressed representations is evaluated by way of relative port current matrix errors, defined as

$$\text{Relative error} = \frac{\left\| I_{\text{port}}^{\text{approximate}} - I_{\text{port}}^{\text{reference}} \right\|_F}{\left\| I_{\text{port}}^{\text{reference}} \right\|_F}. \tag{24}$$

Reference port current matrices $I_{\text{port}}^{\text{reference}}$ are obtained by calculating all mutual inductance matrix entries with the $\varepsilon_{\text{quad}} < 10^{-6}$ recipe, for a direct solution in the case of

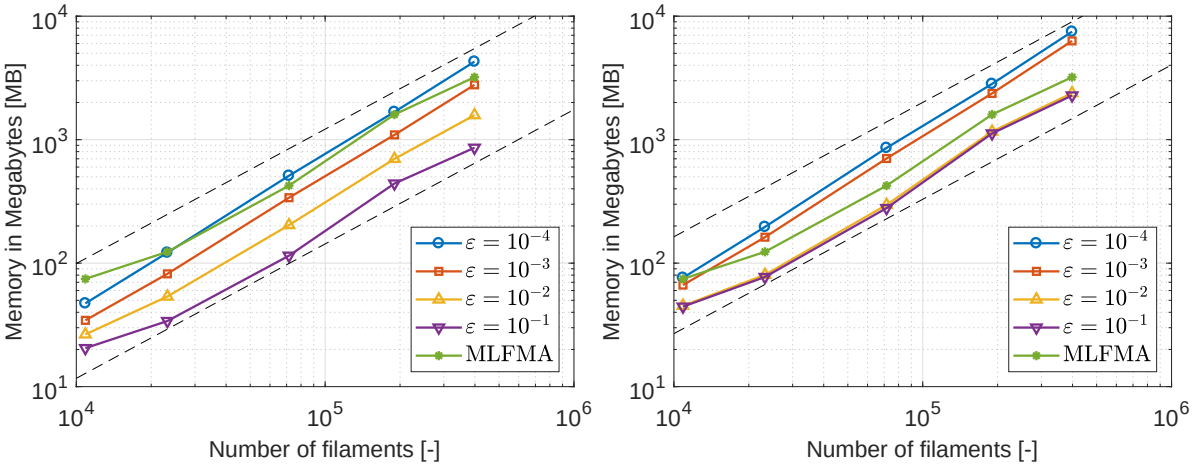

Figure 8. Compressed matrix memory requirement vs. number of filaments, for the AQFP gate. The total number of filaments is varied from 10, 874 to 398, 318; tolerance values refer to ε_{SVD}. Left: MLACA-SVD with nearest-neighbour criterion. Right: MLACA-SVD with second-nearest neighbour criterion.

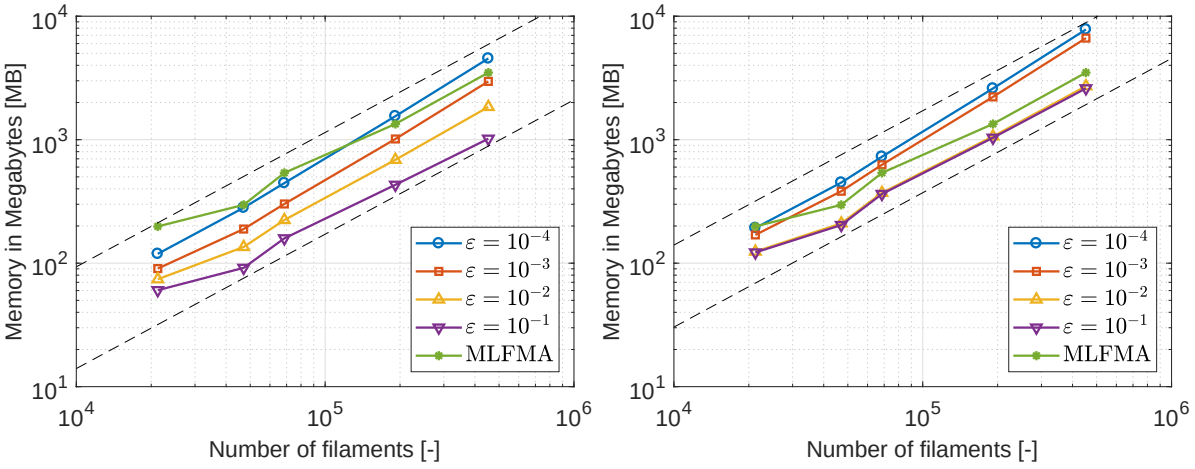

Figure 9. Compressed matrix memory requirement vs. number of filaments, for the eSFQ circuit. The total number of filaments is varied from 21, 251 to 452, 874; tolerance values refer to ε_{SVD}. Left: MLACA-SVD with nearest-neighbour criterion. Right: MLACA-SVD with second-nearest neighbour criterion.

the AQFP gate (23, 226 filaments) and for a second-nearest neigbour criterion, MLACA solution ($\varepsilon_{\text{ACA}} = 10^{-7}$, no SVD recompression) in the case of the eSFQ circuit (38, 895 filaments). The latter solution together with the approximate solutions $I_{\text{port}}^{\text{approximate}}$ (with MLACA-SVD or MLFMA) are obtained with the iterative solver convergence criterion set to an extremely small value, such that the error is only determined by the accuracy of the compressed matrix representation (i.e. these are essentially direct solutions with the reconstituted matrices).

Figure 11 shows the results. Relative errors of $\sim 10^{-2}$ are introduced by the MLFMA. This is due to errors in calculating the near-interaction entries, as well as the coarse, midpoint quadrature upon which the compression is based (see (9)) and

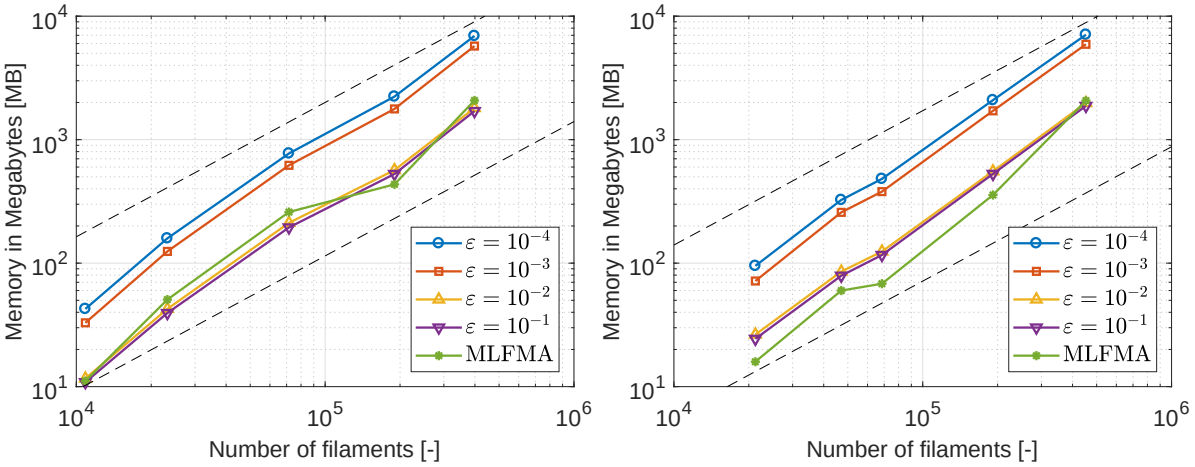

Figure 10. Compressed matrix memory requirement with near-interaction storage excluded, for MLACA-SVD and MLFMA with the same, second-nearest neighbour criterion. Tolerance values refer to ε_{SVD} and $\mathcal{O}(b \log b)$ trend lines are shown. Left: AQFP gate. Right: eSFQ circuit.

the approximate nature of the compression itself. On the other hand, the accuracy with which the MLACA-SVD represents the true matrix is fully controllable through parameter ε_{SVD} (19). An important implication is that when using a mesh which is sufficiently fine to yield a very accurate current distribution solution (when rigorously evaluating the matrix entries), the accuracy can be limited by the MLFMA-introduced error. This is not an issue for the MLACA-SVD, which can in principle solve the electromagnetic field problem to any desired accuracy.

The results of Figure 11 together with those in Figures 8 and 9, indicate that the MLACA-SVD is more efficient than FastHenry's MLFMA. Particularly, the nearest-neighbour criterion together with either $\varepsilon_{\text{SVD}} = 10^{-3}$ or $\varepsilon_{\text{SVD}} = 10^{-2}$ is both more accurate than the MLFMA and require less memory than the MLFMA. For the MLACA-SVD, the nearest-neighbour criterion is clearly more efficient than the second-nearest neighbour criterion, when considering accuracy vs. memory. Nevertheless, the nominal, improved accuracy obtained with the larger near-interaction criterion for the same mesh, does warrant discussion. It is partly due to the nearer interaction criterion simply resulting in a larger percentage of the matrix being approximated [27] (assuming that near interactions are calculated more accurately than compressed ones). However, it is mainly due to the relationship between the η-admissibility condition (23) and accuracy, as explained in Section 5.

7. Conclusion

In this paper an MLACA-SVD solver is presented, which improves the accuracy and efficiency of magnetoquasistatic analysis of superconducting structures with FastHenry, e.g. for superconducting integrated circuit inductance extraction purposes. The MLACA-SVD solver is an alternative to the existing MLFMA solver in FastHenry.

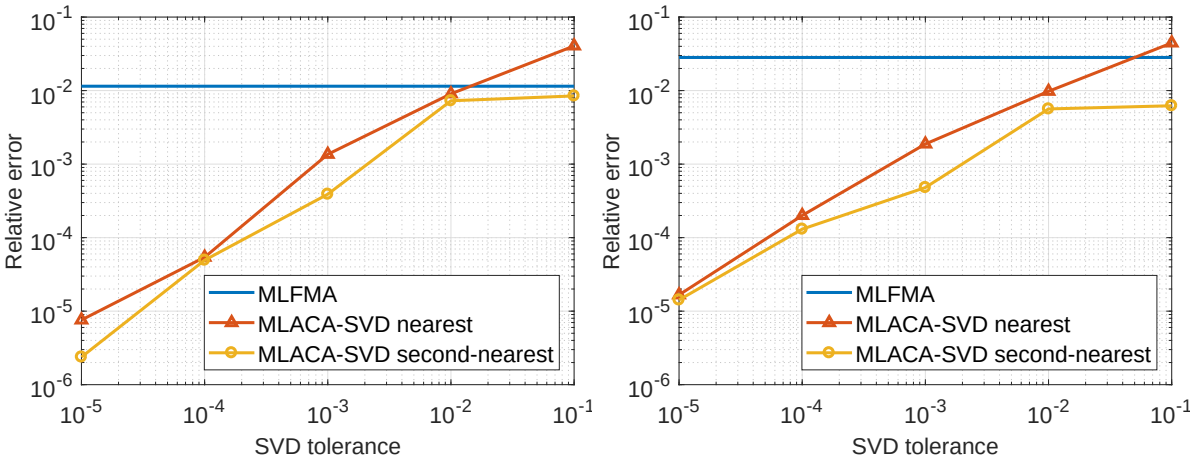

Figure 11. Relative port current matrix errors vs. SVD tolerance. Left: AQFP gate model with 23, 226 filaments. Right: eSFQ circuit model with 38, 895 filaments.

The same underlying set of linear equations is solved, but with different approaches to calculating and storing the near-interaction matrix entries and the compressed, far-interaction matrix entries. Near-interactions are calculated with rigorously determined semi-analytical procedures of which the accuracy can be controlled, rather than using FastHenry's existing, approximate calculations of fixed accuracy. Far interactions are compressed with the MLACA-SVD scheme, which employs the new near-interaction procedures, to evaluate the selected matrix entries needed for compression. For the compression itself, the SVD error tolerance can be set, which controls its accuracy at the cost of increased memory consumption. Two near-interaction distance criteria are tested for the MLACA-SVD, one being exactly the same as that of FastHenry's MLFMA (second-nearest neighbours) and the other being the nearest-neighbour criterion. The latter is found to yield more efficient results.

The MLACA-SVD memory requirement for the compressed far interactions vs. number of filaments, is shown to scale practically identical to that of the MLFMA, for the complex and realistic superconducting integrated circuit models considered. The total memory requirement for the MLACA-SVD solver is lower than that of FastHenry's MLFMA, for the same relative error in the final output (port currents). Beyond this important property, the MLACA-SVD provides full user control over approximation errors in the matrix representation, such that solutions can be obtained to any desired accuracy.

The matrix setup time for the MLACA-SVD is longer than that of FastHenry's MLFMA, due to the former evaluating matrix entries more rigorously, as well as due to algorithmic differences [28]. However, the quadrature recipes could be further optimized (as noted in Section 4) and more generally, the MLACA-SVD matrix setup time can be readily reduced through parallelization, to which the scheme is well suited.

Acknowledgements

This paper is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

The National Research Foundation of South Africa also supported this work (Grant Number 96222).

References

[1] Bunyk P I and Rylov S V 1993 Automated calculation of mutual inductance matrices of multilayer superconductor intergrated circuits *Extended Abstracts of Int. Superconductive Electronics Conf. (ISEC'93) Volume 62*

[2] Fourie C J 2015 Full-gate verification of superconducting integrated circuit layouts with InductEx *IEEE Transactions on Applied Superconductivity* **25**(1) 1300209

[3] Khapaev M M, Kidiyarova-Shevchenko A Y, Magnelind P and Kupriyanov M Y 2001 3D-MLSI: software package for inductance calculation in multilayer superconducting integrated circuits *IEEE Transactions on Applied Superconductivity* **11**(1) 1090–1093

[4] Khapaev M, Kupriyanov M Y, Goldobin E and Siegel M 2003 Current distribution simulation for superconducting multi-layered structures *Superconductor Science and Technology* **16**(1) 24–27

[5] Jackman K and Fourie C J 2016 Tetrahedral modelling method for inductance extraction of complex 3D superconducting structures *IEEE Transactions on Applied Superconductivity* **26**(3) 0602305

[6] Kamon M, White J K and Tsuk M J 1994 FASTHENRY: A Multipole-Accelerated 3-D Inductance Extraction Program *IEEE Transactions on Microwave Theory and Techniques* **42**(9) 1750–1758

[7] Nabors K and White J 1991 FastCap: A Multipole Accelerated 3-D Capacitance Extraction Program *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **10**(11) 1447–1459

[8] Bebendorf M 2000 Approximation of boundary element matrices *Numerische Mathematik* **86**(4) 565–589

[9] Kurz S, Rain O and Rjasanow S 2002 The adaptive cross-approximation technique for the 3-D boundary-element method *IEEE Transactions on Magnetics* **38**(2) 421–424

[10] Bebendorf M and Rjasanow S 2003 Adaptive low-rank approximation of collocation matrices *Computing* **70**(1) 1–24

[11] Zhao K, Vouvakis M N and Lee J F 2005 The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems *IEEE Transactions on Electromagnetic Compatibility* **47**(4) 763–773

[12] Grasedyck L 2005 Adaptive recompression of $\mathcal{H}$-matrices for BEM *Computing* **74**(3) 205–223

[13] Bebendorf M and Kunis S 2009 Recompression techniques for adaptive cross approximation *Journal of Integral Equations and Applications* **21**(3) 331–357

[14] Phillips J R and White J K 1997 A precorrected-FFT method for electrostatic analysis of complicated 3-D structures *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **16**(10) 1059–1072

[15] Yucel A C, Georgakis I P, Polimeridis A G, Bağcı H and White J K 2018 VoxHenry: FFT-accelerated inductance extraction for voxelized geometries *IEEE Transactions on Microwave Theory and Techniques* **66**(4) 1723–1735

[16] Orlando T and Delin K A 1991 *Foundation of Applied Superconductivity* (Addison-Wesley)

[17] Saad Y and Schultz M H 1986 GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems *SIAM Journal on Scientific and Statistical Computing* **7**(3) 856–869

[18] Rjasanow S and Steinbach O 2007 *The fast solution of boundary integral equations* (Springer Science & Business Media)

[19] Grasedyck L and Hackbusch W 2003 Construction and arithmetics of $\mathcal{H}$-matrices *Computing* **70**(4) 295–334

[20] Bebendorf M and Grzhibovskis R 2006 Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation *Mathematical Methods in the Applied Sciences* **29**(January) 1721–1747

[21] Frederix K and Van Barel M 2010 Solving a large dense linear system by adaptive cross approximation *Journal of Computational and Applied Mathematics* **234**(11) 3181–3195

[22] Golub G H and Van Loan C F 1996 *Matrix Computations* (Baltimore: Johns Hopkins University Press) 3rd edn.

[23] Davis P J and Rabinowitz P 1984 *Methods of Numerical Integration* (Orlando, Florida: Academic Press) 2nd edn.

[24] Wilton D R, Rao S M, Glisson A W, Schaubert D H, Al-Bundak M and Butler C M 1984 Potential Integrals for Uniform and Linear Source Distributions on Polygonal and Polyhedral Domains *IEEE Transactions on Antennas and Propagation* **32**(3) 276–281

[25] Takeuchi N, Ortlepp T, Yamanashi Y and Yoshikawa N 2014 Novel latch for adiabatic quantum-flux-parametron logic *Journal of Applied Physics* **115**(10) 103910

[26] Vernik I V, Kaplan S B, Volkmann M H, Dotsenko A V, Fourie C J and Mukhanov O A 2014 Design and test of asynchronous eSFQ circuits *Superconductor Science and Technology* **27**(4) 044030

[27] Tamayo J M, Heldring A and Rius J M 2011 Multilevel adaptive cross approximation (MLACA) *IEEE Transactions on Antennas and Propagation* **59**(12) 4600–4608

[28] Brunner D, Junge M, Rapp P, Bebendorf M and Gaul L 2010 Comparison of the fast multipole method with hierarchical matrices for the Helmholtz-BEM *Computer Modeling in Engineering & Sciences (CMES)* **58**(2) 131–160

Appendix D

Journal paper – MLACA with modified grouping [10]

B. A. P. Nel and M. M. Botha, "MLACA with modified grouping strategy for efficient superconducting circuit analysis," in Applied Superconductivity Conference (ASC), Seattle, United States of America, October 2018, 4 pages.

MLACA with Modified Grouping Strategy for Efficient Superconducting Circuit Analysis

Ben A. P. Nel and Matthys M. Botha, *Member, IEEE*

***Abstract*—Electromagnetic analysis of superconducting integrated circuits is routinely required for inductance extraction. FastHenry is a well-known, applicable magnetoquasistatic analysis tool. It is accelerated with the multilevel fast multipole algorithm (MLFMA). A multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) was recently proposed for FastHenry, as an alternative to the MLFMA. It was found to be more efficient, with regards to accuracy versus required memory. It uses the same octree grouping as FastHenry's MLFMA. Here, two modified grouping strategies are proposed to further improve MLACA-SVD efficiency by compressing interactions between larger groups, while maintaining scaling performance consistent with valid admissibility of interactions. The strategies are denoted 'shell merging' and 'wall merging'. Results show that the MLACA-SVD solver with merging requires about four times less memory than FastHenry's MLFMA, for similar accuracy. Shell merging is found to be slightly superior.**

***Index Terms*—Group interaction, hierarchical octree, low-rank factorization, partial element equivalent circuit (PEEC), volume integral equation.**

I. Introduction

INDUCTANCE extraction for superconducting integrated circuits requires the accurate solution of structural current distributions [1]–[4]. This task can be accomplished with FastHenry, which is a well-known, integral equation-based, magnetoquasistatic analysis tool [5]. Storing the entire, dense inductance matrix of dimensions $b \times b$, where b denotes the number of mesh filaments, is prohibitively expensive in computational terms. Therefore, FastHenry uses the multilevel fast multipole algorithm (MLFMA) to compress this matrix. Multilevel adaptive cross approximation with singular value decomposition recompression (MLACA-SVD) was proposed in [6]–[10], for compression of boundary element method (BEM) matrices with asymptotically smooth kernels. Recently, an MLACA-SVD solver was presented as an alternative to the MLFMA, within FastHenry [11]. Results for circuit structures of practical interest show that the MLACA-SVD solver is more efficient than FastHenry's MLFMA. Furthermore, the MLACA-SVD solver offers comprehensive control over the errors introduced by the algorithm. In [11] quadrature recipes of guaranteed accuracy are presented, such that matrix-entry numerical integration error contributions are small enough that overall accuracy is solely determined by the compression tolerance setting. The MLACA-SVD solver compresses off-diagonal matrix blocks representing interactions between groups of observer and source filaments. This is done on each level. Standard, hierarchical, octree grouping for three-dimensional meshes is employed according to [12]. In pursuit of further matrix storage reduction, group merging strategies for compression of larger matrix blocks at a time, are proposed in this paper. Such strategies would be very difficult to realize within an MLFMA solver, while being fairly straightforward for the MLACA-SVD due to its algebraic nature.

Section II reviews the MLACA-SVD solver and provides motivation for the merging of groups, based on the spherically-symmetric nature of the Green's function and the ACA's group interaction admissibility condition. Two group-merging strategies are presented in Sections III and IV, denoted *shell merging* and *wall merging*, respectively. Section V presents numerical results to assess the performance of the modified grouping strategies, for superconducting integrated circuit modeling. Section VI concludes the work.

II. MLACA-SVD Overview and Motivation for Group Merging

Circuit structures are discretized into hexahedral filaments with constant, axial current densities. Superconductivity is incorporated via the filaments' self resistances [11]. Magnetic coupling is incorporated via the mutual inductance matrix L [5], [11]. A matrix entry representing the coupling between observer (i.e. testing) filament i and source filament j, is calculated as

$$L_{ij} = \frac{\mu}{4\pi a_i a_j} \int_{V_i} \int_{V_j} \frac{\hat{\ell}_i \cdot \hat{\ell}_j}{|\boldsymbol{r} - \boldsymbol{r}'|} dV' dV, \tag{1}$$

where a_i, V_i and $\hat{\ell}_i$ denote the cross-sectional area, volume, and axial unit vector of filament i, respectively. As shown in [11], the axial unit vectors are factorized out from L in a straightforward manner, leaving the *potential matrix* for MLACA-SVD compression. A near-interaction criterion determines if an observer filament group P and source filament group Q on a given level of the octree, are 'near' or 'far'; near interactions are handled down-level, while all far

B. A. P. Nel and M. M. Botha are with the Department of Electrical and Electronic Engineering, Stellenbosch University, South Africa (e-mail: 17762944@sun.ac.za, mmbotha@sun.ac.za).

This paper is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

The National Research Foundation of South Africa also supported this work (Grant Number 96222).

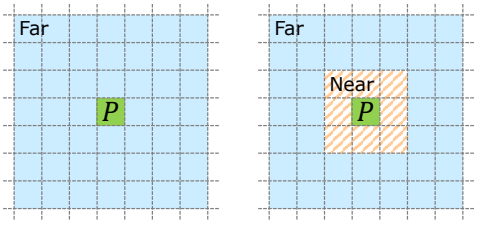

Fig. 1. Two near-interaction criteria, with respect to observer group P. Left: self interaction criterion. Right: nearest-neighbor criterion.

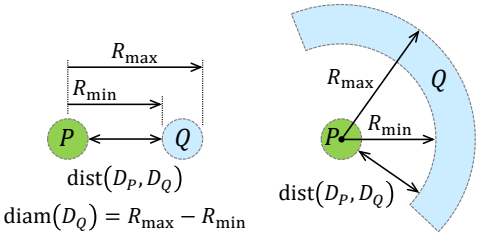

Fig. 2. Left: alternative method of group diameter measurement. Right: set-up with a non-convex source group, for which the alternative η-admissibility condition is identical to that of the set-up on the left.

interactions are compressed. Fig. 1(right) shows the nearest-neighbor criterion used throughout this paper, unless stated otherwise.

The actual suitability for compression of an off-diagonal matrix block representing the coupling between observer group P and source group Q, is governed by admissibility conditions. The strong η-admissibility condition is as follows [6], [13]:

$$\max\{\mathrm{diam}(D_P), \mathrm{diam}(D_Q)\} \leq \eta\, \mathrm{dist}(D_P, D_Q), \qquad (2)$$

where D_P and D_Q are the smallest possible, convex hulls enclosing each of the two filament groups. It is required that $\eta > 0$ and be bounded. Fixing the value of η limits the extent to which the kernel function $1/|\boldsymbol{r} - \boldsymbol{r}'|$ varies over the two domains. Consequently, it determines the extent of achievable low-rank compression to a given accuracy, for the off-diagonal matrix block; this relates to the nature of exponential decay in the block's singular values.

Now allow for the enclosing domains D_P and D_Q in (2) to be non-convex, together with an alternative way of measuring diameter, namely from the perspective of any given point inside the other domain, as shown in Fig. 2(left). This yields an alternative η-admissibility condition. For the set-up in Fig. 2(left), both the conventional and alternative admissibility conditions yield the same η value. However, for the set-up in Fig. 2(right) the conventional approach yields a dramatically larger (i.e. implied worse) value, while the alternative approach yields the same value as before. The fact that the alternative approach yields equal values, speaks to the kernel function's actual range of variation being equal in both cases, which is due to its spherically-symmetric nature.

The above discussion suggests that a set of observer groups and a set of source groups that are all interacting on a given octree level, could be selected such that the alternative η-admissibility condition for the two merged groups is fairly similar to those of the individual interactions. Thus, minimized increases in the alternative η values, which are expected to minimally reduce compression efficiency, can be traded for compressing larger matrix blocks, which is generally beneficial. Two practical merging strategies are presented next.

III. Shell Merging

In both merging strategies, each group on a given octree level is considered as observer in turn, to compress all of its far interactions with source groups as well as some of its neighbors' interactions with some of these same source groups. The last observer is not considered, since by definition its interactions would have already been dealt with. For merging purposes the observer group under current consideration is labelled the *anchor group*.

Shell merging is explained with the aid of the example in Fig. 3. Starting with a given anchor, three of its direct neighbors (sharing at least a vertex with it) are sought, such that these neighbors each also shares at least a vertex with one of the others, to ensure a tight cluster with minimized diameter. This yields a merged observer group consisting of four octree groups. The set of far-interaction source groups that are shared by all four observer groups, which can sometimes resemble a shell around the observer cluster, is then identified as shown in Fig. 3(a). These source groups are merged and the interaction between merged observer and source groups is compressed with ACA-SVD. Next, three other anchor neighbors are sought according to the same procedure, and the process is repeated, as shown in Fig. 3(b). If there are not enough neighbors left or if three with any shared source interactions cannot be found, then two neighbors are sought instead, as shown in Figs 3(c) and 3(d). These may now include some nearest neighbors previously merged into observer clusters of four, since the condition of shared source interactions is different when the observer cluster is smaller. In the case of the example, all anchor interactions were covered after four steps. In general, the process can progress to seeking only a single neighbor to merge with the anchor; and ultimately to compress the interaction between the anchor on its own, with the merged set of all its remaining far-interaction source groups.

Note that as more and more anchors are processed, less merging opportunities will remain and consequently the large merges shown in the example of Fig. 3 become more scarce.

Finally, the starting goal of merging four observers (rather than a higher number) is chosen for two reasons: it ensures a fair chance that these observers have many far interactions with common source groups, especially when all observers belong to the same octree parent; and it ensures that the merged group diameter is not too large, which would increase η.

IV. Wall Merging

Wall merging entails a fixed number of six steps for each anchor group, as shown by the example in Fig. 4. The anchor group's far-interaction source groups are divided into six

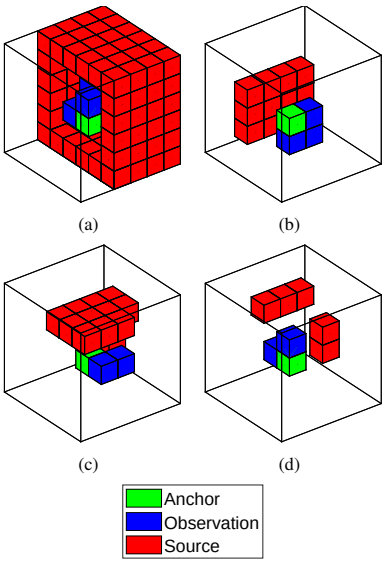

Fig. 3. Shell merging example, which shows the four steps (a) to (d), required in this case to cover all of the anchor's far interactions with source groups. The coloured cubes represent octree groups and the large wire-frame cube depicts the bounding box of all of the anchor's far-interaction source groups. Note that in (a), some of the shared-interaction source groups are not shown for clarity (i.e. a third of the merged "shell" is cut away).

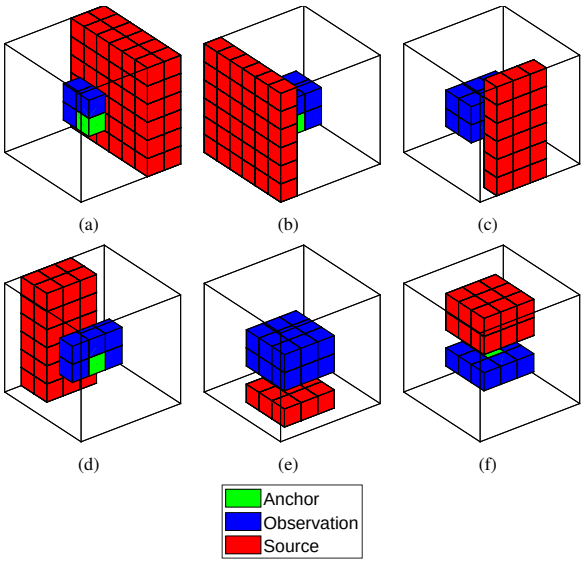

Fig. 4. Wall merging example, which shows the six steps (a) to (f), required to cover all of the anchor's far interactions with source groups. The coloured cubes represent octree groups and the large wire-frame cube depicts the bounding box of all of the anchor's far-interaction source groups.

merged clusters, for compression. Each cluster relates to a face of the anchor group's cubic shape. A cluster is obtained as all of the anchor's remaining far-interaction source groups, which lie beyond a flat plane associated with a face. This plane is defined to be parallel to the face and placed at a normal distance of one group side-length away from the face. The source cluster can resemble a wall, as shown in Figs 4(a) to 4(f). Observer groups to be merged with the anchor, are determined as all of its direct neighbors which must still interact with the whole of the merged source. The merged observer cluster sometimes also resembles a wall. Note that unlike with shell merging, it can happen with wall merging that the same merged observation cluster is used multiple times. As with shell merging, fewer merging opportunities will remain as the number of processed anchors increase.

V. Results

Fig. 5, generated using InductEx [14], shows the superconducting integrated circuit test model [15], which represents an inductance test SQUID. It is meshed with varying filament numbers to obtain all of the results.

Consider a fixed mesh with $24,272$ filaments. The SVD error tolerance $\varepsilon_{\mathrm{SVD}}$, which controls the MLACA-SVD matrix compression error [11], is varied. Fig. 6 compares the normal MLACA-SVD, shell-merging MLACA-SVD and wall-merging MLACA-SVD, with FastHenry's MLFMA (the latter gives fixed results, because $\varepsilon_{\mathrm{SVD}}$ is not relevant to it). Observe in Fig. 6(top left) that the normal MLACA-SVD's dependence of memory requirement upon the logarithm of $1/\varepsilon_{\mathrm{SVD}}$ is preserved by the merging versions. For a fixed value of $\varepsilon_{\mathrm{SVD}}$, both merging versions yield a fixed reduction in required memory of approximately 30%, with wall merging being slightly superior. A fifth trace is also shown, for normal MLACA-SVD with the self near-interaction criterion of Fig. 1(left). This criterion implies $\eta = \infty$ in (2), which is not a valid condition. Its memory scaling is dramatically poorer than the others. The inclusion of this poor result serves to emphasize that the two merging strategies are consistent with maintained admissibility. The solution accuracy results shown in Fig. 6(top right) are the relative errors in the port currents, as defined in [11]. These results show that convergence towards the true mutual inductance matrix as compression tolerance is decreased, is fairly similar for the normal MLACA-SVD and the two merging versions, especially for $\varepsilon_{\mathrm{SVD}} \leq 10^{-3}$. For a fixed value of $\varepsilon_{\mathrm{SVD}}$, shell merging generally yields superior accuracy to wall merging, which is ascribed to the geometric properties of the former (see Fig. 3) more closely adhering to the idea of maintained (alternative) admissibility as expressed in Fig. 2. Fig. 6(bottom) combines the two result sets. This shows that for the problem at hand, shell merging is slightly more efficient than wall merging and that both are more efficient than normal MLACA-SVD. All MLACA-SVD versions outperform FastHenry's MLFMA.

Now consider memory scaling with respect to the number of filaments, as shown in Fig. 7. The merging versions scale exactly like the normal MLACA-SVD. Furthermore, the reduction by approximately 30% relative to the normal MLACA-SVD is affirmed for both merging versions, with wall merging again being slightly superior when only memory is considered. Note that for $\varepsilon_{\mathrm{SVD}} = 10^{-2}$, where the merging versions yield compression accuracies similar to FastHenry's MLFMA, the memory requirement is less by about a factor of four.

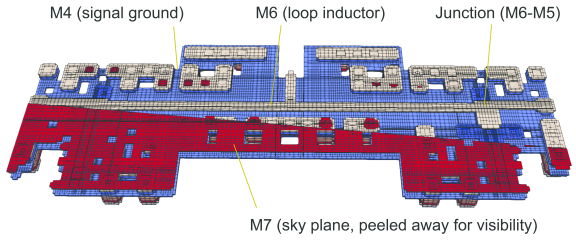

Fig. 5. Test model representing an inductance test SQUID for experimental measurements, with loop inductance in M6 sandwiched between a ground plane in M4 and a ground-connected sky plane in M7 [15].

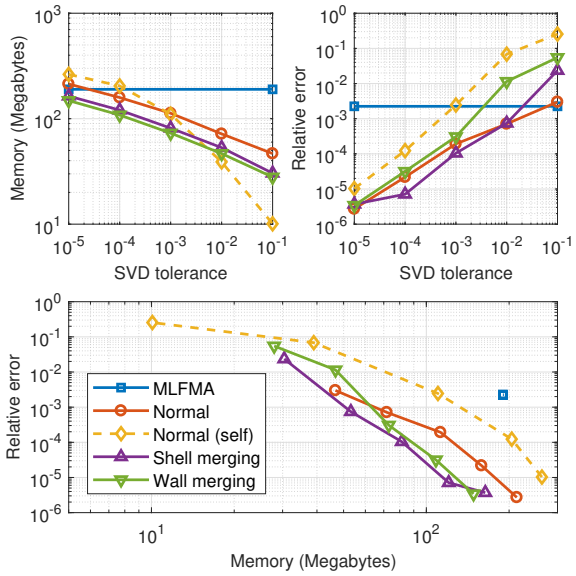

Fig. 6. Memory required to store the mutual inductance matrix (top left) and relative error in the solution port currents (top right). Both quantities are shown as functions of the matrix compression error tolerance $\varepsilon_{\mathrm{SVD}}$, for a mesh of $24,272$ filaments. At the bottom, the relative error is shown as a function of memory.

VI. Conclusion

With the objective of efficient electromagnetic analysis of superconducting integrated circuits, an MLACA-SVD solver was recently presented for the well-known FastHenry package [11]. It was found to be more efficient than FastHenry's MLFMA solver, with regards to accuracy versus required memory. In this paper, group merging is introduced as a method to further improve the efficiency of this MLACA-SVD solver. Motivation for the concept of merging is provided on the basis of η-admissibility considerations. Two group merging strategies are proposed. Taking the analysis of a SQUID as an example, numerical results show that both merging strategies yield results consistent with valid admissibility. They reduce the required memory of the MLACA-SVD by approximately 30%. Shell merging is marginally more efficient than wall merging, with regards to accuracy versus required memory. For results of similar accuracy as FastHenry's MLFMA, the

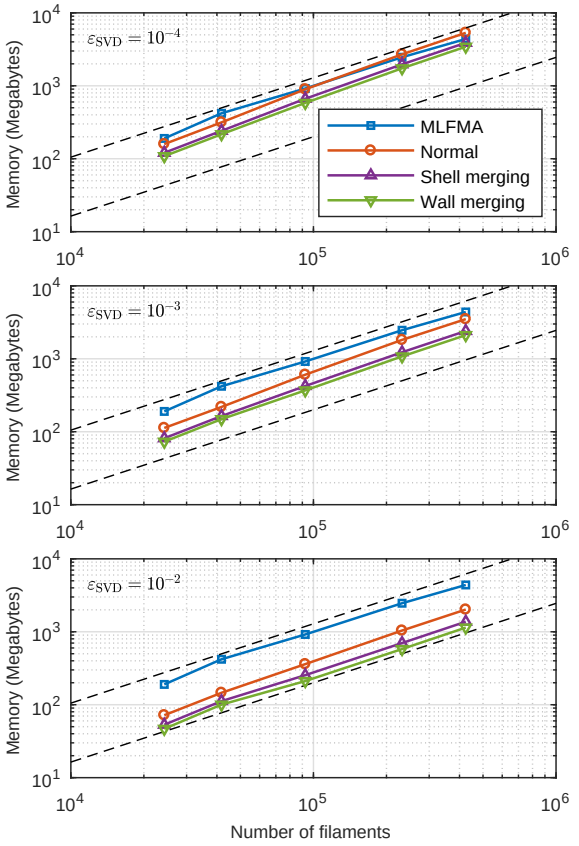

Fig. 7. Memory required to store the mutual inductance matrix, versus number of filaments b. The dashed trend lines indicate $\mathcal{O}(b \log b)$ scaling. Top: results with $\varepsilon_{\mathrm{SVD}} = 10^{-4}$. Middle: results with $\varepsilon_{\mathrm{SVD}} = 10^{-3}$. Bottom: results with $\varepsilon_{\mathrm{SVD}} = 10^{-2}$.

MLACA-SVD solver with merging requires about four times less memory. For general volumetric structures, rather than the planar integrated circuits of interest here, merging could possibly yield even larger memory reductions due to more opportunities for it. These merging strategies could be tested for other ACA solvers featuring the dynamic kernel, e.g. [16].

References

[1] P. I. Bunyk and S. V. Rylov, "Automated calculation of mutual inductance matrices of multilayer superconductor intergrated circuits," in *Extended Abstracts of Int. Superconductive Electronics Conf. (ISEC'93)*, vol. 62, 1993.

[2] M. M. Khapaev, A. Y. Kidiyarova-Shevchenko, P. Magnelind, and M. Y. Kupriyanov, "3D-MLSI: software package for inductance calculation in multilayer superconducting integrated circuits," *IEEE Transactions on Applied Superconductivity*, vol. 11, no. 1, pp. 1090–1093, 2001.

[3] M. Khapaev, M. Y. Kupriyanov, E. Goldobin, and M. Siegel, "Current distribution simulation for superconducting multi-layered structures," *Superconductor Science and Technology*, vol. 16, no. 1, p. 24, 2002.

[4] K. Jackman and C. J. Fourie, "Tetrahedral modelling method for inductance extraction of complex 3D superconducting structures," *IEEE Transactions on Applied Superconductivity*, vol. 26, no. 3, pp. 1–5, 2016.

[5] M. Kamon, J. K. White, and M. J. Tsuk, "FASTHENRY: A Multipole-Accelerated 3-D Inductance Extraction Program," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 9, pp. 1750–1758, 1994.

[6] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Mathematik*, vol. 86, no. 4, pp. 565–589, 2000.
[7] S. Kurz, O. Rain, and S. Rjasanow, "The adaptive cross-approximation technique for the 3-D boundary-element method," *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 421–424, 2002.
[8] M. Bebendorf and S. Rjasanow, "Adaptive low-rank approximation of collocation matrices," *Computing*, vol. 70, no. 1, pp. 1–24, 2003.
[9] L. Grasedyck, "Adaptive recompression of $\mathcal{H}$-matrices for BEM," *Computing*, vol. 74, no. 3, pp. 205–223, 2005.
[10] M. Bebendorf and S. Kunis, "Recompression techniques for adaptive cross approximation," *Journal of Integral Equations and Applications*, vol. 21, no. 3, pp. 331–357, 2009.
[11] B. A. P. Nel and M. M. Botha, "An efficient MLACA-SVD solver for superconducting integrated circuit analysis," 2018, submitted for publication.
[12] K. Nabors and J. White, "FastCap: A Multipole Accelerated 3-D Capacitance Extraction Program," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 11, pp. 1447–1459, 1991.
[13] L. Grasedyck and W. Hackbusch, "Construction and arithmetics of $\mathcal{H}$-matrices," *Computing*, vol. 70, no. 4, pp. 295–334, 2003.
[14] C. J. Fourie, InductEx, Stellenbosch Univ., Stellenbosch, South Africa, 2018. [Online]. Available: http://www.inductex.info.
[15] C. J. Fourie, C. Shawawreh, I. V. Vernik, and T. V. Filippov, "High-accuracy InductEx calibration sets for MIT-LL SFQ4ee and SFQ5ee processes," *IEEE Transactions on Applied Superconductivity*, vol. 27, no. 2, pp. 1–5, 2017.
[16] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Transactions on Electromagnetic Compatibility*, vol. 47, no. 4, pp. 763–773, 2005.